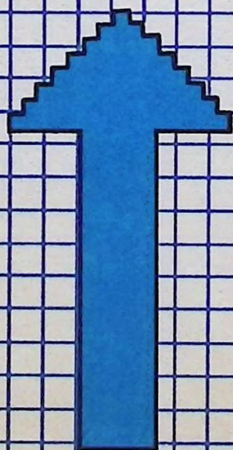
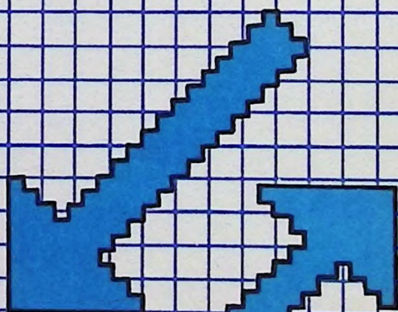


Mikrokomputery

JAK MIERZYĆ STEROWAĆ I REGULOWAĆ za pomocą Basicu?

Wolfgang Link





Jak mierzyć, sterować i regulować za pomocą Basicu?

Mikrokomputery

Wolfgang Link

Jak mierzyć, sterować i regulować za pomocą Basicu?

Z języka niemieckiego tłumaczył
WOJCIECH ZAWORSKI



Wydawnictwa Naukowo-Techniczne
Warszawa 1988

Dane o oryginale:

Wolfgang Link

Messen, Steuern und Regeln mit Basic

Programmierung und lauffähige Programme für die elektronische Messtechnik

Opiniodawca *Wojciech Nowakowski*
Redaktor *Ewa Zdanowicz*
Redaktor techniczny *Barbara Chojnacka-Flisiuk*
Okładkę i strony tytułowe projektował *Juliusz Rybicki*

681.3

W książce podano podstawowe informacje, zasady i przykłady z programami dotyczące wykorzystania mikrokomputerów do pomiarów różnych wielkości fizycznych, sterowania urządzeniami elektrycznymi oraz tworzenia automatycznych minisystemów regulacyjnych. Przedstawiono również podstawowe dane najczęściej stosowanych interfejsów do operacji wejścia-wyjścia.
Książka jest przeznaczona dla szerokiego kręgu czytelników — użytkowników popularnych mikrokomputerów.

ISBN 83-204-0934-9

Spis treści

Słowo wstępne

Książka ta nie była pisana z myślą ani o specjalistach z zakresu techniki sterowania i regulacji, ani o handlowcach z tej branży, powinna natomiast wprowadzić amatora w zagadnienie tytułowe i pokazać mu szersze pole zastosowań dla jego własnego komputera.

Przedstawione układy i programy zostały wprawdzie zrealizowane przy zastosowaniu komputera TRS-80[1] firmy Tandy (Radio Shack), ale przy odrobinie wysiłku można — modyfikując zarówno potrzebne interfejsy, jak i programy — przystosować je do innego komputera i innej wersji Basicu.

Świadomie przedstawiono tylko niezbyt złożone rozwiązania układowe, aby początkującemu użytkownikowi zapewnić łatwe zapoznanie się z zagadnieniami automatyzacji pomiarów, sterowania i regulacji dokonywanych przy użyciu komputera. Wówczas zestawienie większej liczby przetworników lub bloków sterujących w jedno stanowisko pomiarowe, ewentualnie w bardziej złożony system, nie powinno już stanowić problemu.

Wolfgang Link, Paderborn

---

[1] Polskim odpowiednikiem tego komputera jest MERITUM-1 produkowany przez zakłady MERA-ELZAB w Zabrzu (przyp. tłum.).

1. Interfejsy

Jeśli trzeba połączyć z komputerem urządzenia zewnętrzne (peryferyjne), jak np. przekaźniki, silniki, wskaźniki, czujniki pomiarowe, programowane zasilacze, generatory częstotliwości itd., to niezbędne jest zastosowanie układów sprzęgających. Ponieważ wymagane przez urządzenia peryferyjne poziomy napięć, prądów i mocy są z reguły różne od tych, które dostarcza komputer, a przede wszystkim ze względu na to, że szybkość działania tych urządzeń różni się znacznie od szybkości działania komputera, potrzebne są specjalne układy, które zapewniają odpowiednie dopasowanie. Układy te noszą nazwę *interfejsów*[1)2)]. Producenci mikrokomputerów oferują specjalne układy interfejsów przystosowane do poszczególnych typów procesorów. Aby dostawcy urządzeń peryferyjnych (np. drukarek, ploterów) nie musieli dostosowywać swoich urządzeń do dużej liczby takich układów pośredniczących, niektóre systemy interfejsów zostały znormalizowane bądź też podniesione do rangi quasi-standardu. Do najbardziej znanych należą: interfejs V24 (w USA zwany RS-232C), system interfejsu IEC-625 zwany krócej magistralą IEC (w USA: IEEE-488) oraz często używany quasi-standardowy interfejs Centronics łączący drukarkę z komputerem osobistym. O ile w przypadku interfejsu V24 dane są przesyłane szeregowo, tzn. bit po bicie, o tyle magistrala IEC zapewnia transmisję równoległą, a więc jednego bajtu danych podczas jednego taktu. Zaletą transmisji szeregowej jest mała liczba połączeń, ale wymagany czas transmisji jest przy tej samej częstotliwości taktowania znacznie dłuższy niż w przypadku przesyłania równoległego. Tu z kolei za szybkość transmisji płaci się liczbą połączeń, z których co najmniej 7 do 8 pełni funkcję linii danych, a dodatkowo potrzeba jeszcze kilku linii sterujących.

---

1) Interfejs (ang. *interface*) – układ pośredniczący między dwoma urządzeniami (przyp. tłum.).
2) Odpowiednikiem polskim interfejsu jest *sprzęg* (przyp. red.).

Mimo że książka ta jest poświęcona głównie przykładowym zastosowaniom prostych interfejsów, opisane zostały w niej również układy standardowe, ponieważ w nowszych mikrokomputerach są one coraz częściej stosowane.

1.1. Interfejsy standardowe

1.1.1. System interfejsu IEC-625[1)]

System interfejsu IEC-625 jest zwany w skrócie magistralą lub szyną IEC (ang. *IEC-Bus*). Wywodzi się on z rozwiązania wprowadzonego przez firmę Hewlett-Packard, które w roku 1975 zostało znormalizowane w USA jako standard IEEE-488, a w roku 1980 jako międzynarodowy standard IEC-625. Oba systemy są w pełni kompatybilne pod względem elektrycznym, a różnią się jedynie typem użytych złączy. Norma IEEE przewiduje użycie 24-stykowego złącza firmy Amphenol, a IEC — 25-stykowego złącza firmy Canon.

W praktyce przeważająca liczba urządzeń używanych obecnie w przemyśle jest wyposażona w złącza 24-stykowe. Na rysunku 1.1 są pokazane oba typy złączy oraz sposób ich okablowania.

W odróżnieniu od interfejsu V24, który pozwala jedynie na wymianę danych między dwoma urządzeniami, magistrala IEC umożliwia współdziałanie wielu różnych urządzeń poprzez wspólną magistralę systemową, określoną w języku ang. jako "Party-Line-System", do której wszystkie te urządzenia są przyłączone.

Magistrala IEC ma następujące cechy użytkowe:

— Do magistrali można przyłączyć maksymalnie 15 urządzeń.

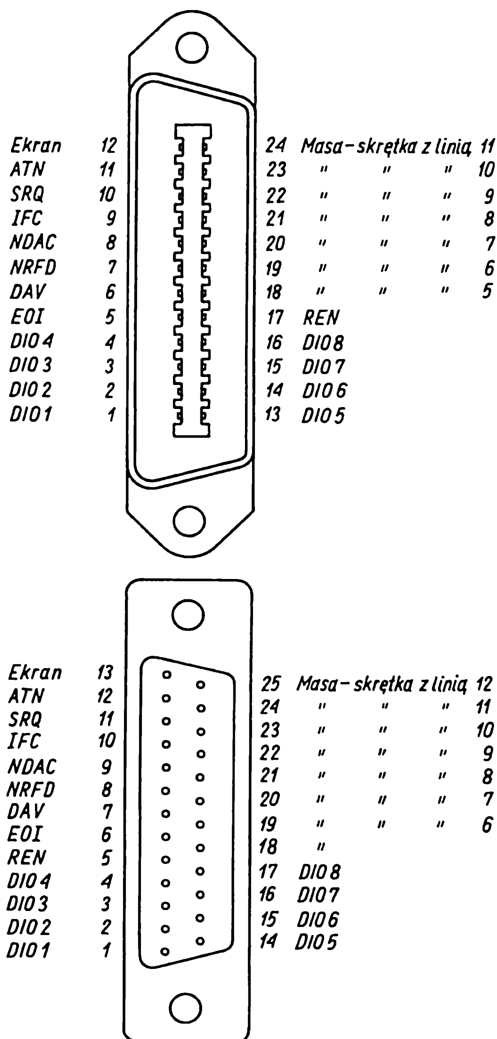
— Długość całego systemu połączeń nie powinna przekraczać 20 m. Przy tej długości transmisja może odbywać się z prędkością 250 do 500 kbajtów na sekundę, przy czym w praktyce jest ona określona przez przyłączone urządzenia. Odległość między sąsiednimi urządzeniami nie powinna przy tym przekraczać czterech metrów.

— Magistrala systemu składa się z 16 linii: 8 linii danych i 8 linii sterujących.

— Transmisja informacji odbywa się szeregowo-równolegle, tzn. bitowo-równolegle i bajt po bajcie — szeregowo. Stosowany jest przy tym kod ISO-7, którego 7 bitów informacyjnych uzupełniane jest bitem parzystości.

— Wszystkie linie są kompatybilne ze standardem TTL, a wszystkie sygnały są aktywne w stanie niskim.

[1)] Dokładniejszy opis działania systemu IEC-625 znajdzie czytelnik w książce: W. Nowakowski i in. *System interfejsu IEC-625*, WKiŁ, Warszawa 1984 (przyp. tłum.).



Rys. 1.1. Okablowanie złącza IEEE-488 (a); okablowanie złącza IEC-625 (b) — w obu przypadkach widok z przodu

Urządzenia połączone ze sobą w jeden system można podzielić na cztery rodzaje:

— Odbiorca (ang. *listener*) — może jedynie odbierać dane.

— Nadawca (ang. *talker*) — może jedynie wysyłać dane.

— Nadawca/odbiorca — może, zależnie od wyboru, wysyłać lub odbierać dane.

— Kontroler (ang. *controller*) — jednostka sterująca np. komputer.

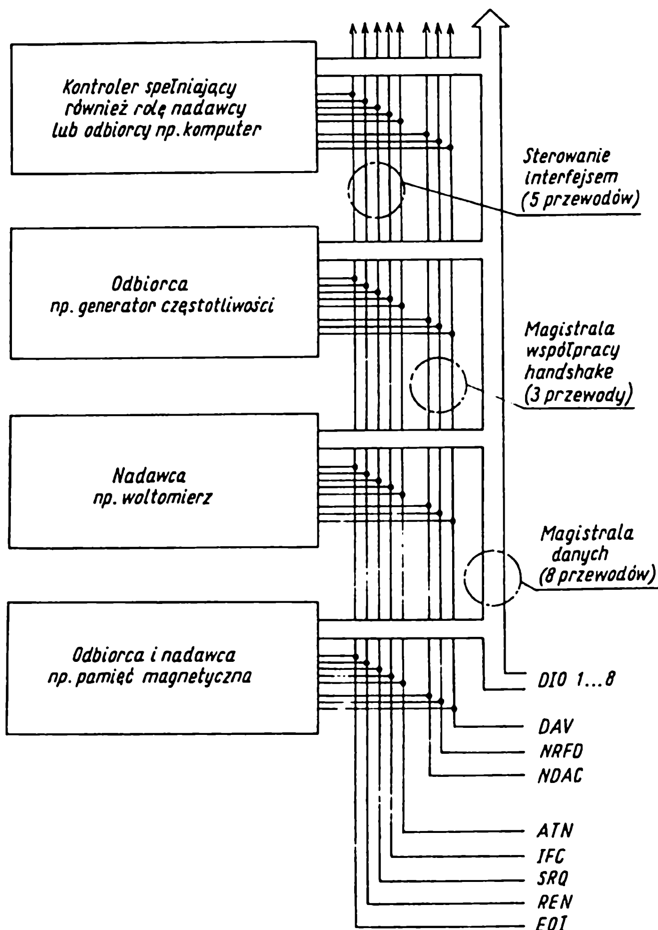
Potrzebny jest tylko jeden kontroler. Jego zadaniem jest adresowanie urządzeń biorących udział w wymianie informacji oraz sterowanie wymianą danych.

Na rysunku 1.2 jest pokazana struktura magistrali interfejsu. Jak można zauważyć, osiem linii sterujących podzielono na dwie grupy: pierwsza grupa to pięć linii sterowania interfejsów, pozostałe trzy służą do kontroli transmisji (ang. *handshake*). Są to:

— DAV (ang. *Data Valid* — dane ważne)

Nadawca informuje, że na liniach DIO jest nowy bajt danych.

— NRFD (ang. *Not Ready For Data* — nie gotowy do przyjęcia



Rys. 1.2. Struktura magistrali IEC

danych). Nie wszystkie urządzenia są gotowe do odbioru danych. Przesyłanie danych może rozpocząć się dopiero wówczas, gdy wszystkie urządzenia (wytypowane jako odbiorcy) zgłoszą swoją gotowość. Dzięki równoległemu połączeniu wyjść NRFD wszystkich urządzeń tworzy się funkcja AND (ang. *Wired And).*

— NDAC (ang. Not Data Accepted — dane nie przyjęte)
Jeszcze nie wszystkie urządzenia przyjęły dane.

— ATN (ang. *Attention* — uwaga)
Zero (linia w stanie wysokim — H) oznacza, że na liniach DIO znajdują się dane (tryb przesyłania danych). Jedynka na tej linii (stan L) sygnalizuje, że poprzez linie danych są przesyłane adresy albo rozkazy.

— IFC (ang. *Interface Clear* — zerowanie interfejsu)
Linię tę obsługuje jedynie kontroler. Służy ona do tego, aby wszystkie przyłączone do magistrali urządzenia wprowadzić w określony stan początkowy (np. po włączeniu napięcia sieciowego).

— SRQ (ang. *Service Request* — żądanie obsługi)
Sygnał SRQ pojawia się wówczas, gdy jedno lub więcej z przyłączonych do magistrali urządzeń żąda obsługi. Kontroler musi wówczas stwierdzić, które z urządzeń zgłosiło żądanie, np. w ten sposób, że sprawdza po kolei każde z nich.

— REN (ang. *Remote Enable* — sterowanie zdalne)
Sygnał REN przygotowuje połączone z magistralą urządzenia do zdalnego sterowania. Jeśli któreś z nich zostało zaadresowane jako odbiorca, to jego elementy regulacyjne na płycie czołowej zostaną wyłączone.

— EOI (ang. *End Or Identify* — koniec lub identyfikacja)
Linia EOI pełni dwie różne funkcje zależnie od tego, czy mamy do czynienia z trybem przesyłania danych, czy też jest to tryb przesyłania instrukcji (por. sygnał ATN). W pierwszym przypadku linia EOI sygnalizuje, że na magistrali danych znajduje się bajt kończący transmisję. Jeśli urządzenie znajduje się w trybie instrukcji, to stan aktywny linii EOI oznacza, że rozpoczęło się sprawdzanie, które z przyłączonych urządzeń zażądało obsługi (ang. *service request*).

Przebieg współpracy

Opisany zostanie teraz przebieg współpracy (ang. *handshake*) podczas przesyłania danych i instrukcji.

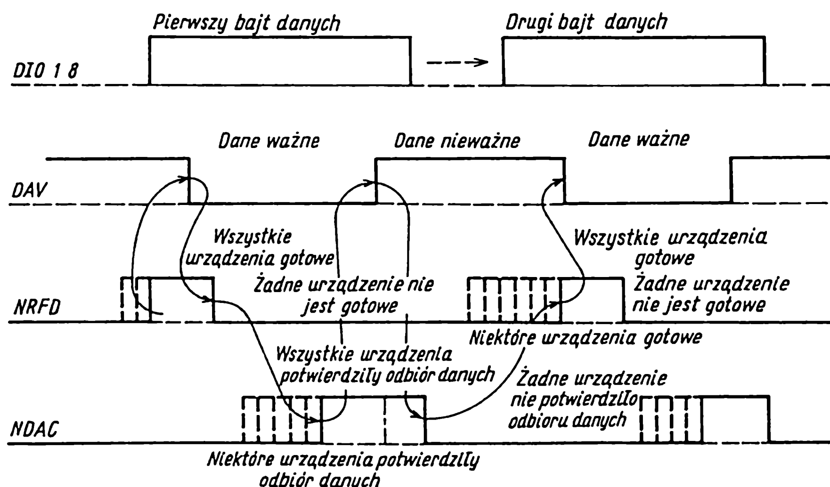
Przed rozpoczęciem przesyłania danych, kontroler wyznacza, przez wysłanie odpowiednich adresów, jedno lub więcej urządzeń, które będą pełnić funkcję odbiorców (ang. *listener*), oraz jedno urządzenie jako nadawcę (ang. *talker*). Adresy urządzeń musi ustalić użytkownik w chwili przyłączania ich do magistrali interfejsu. Do tego celu służy zazwyczaj pięć miniaturowych przełączników, za pomocą których ustawia się 5-bitowy adres. Ponieważ adres 31 jest zajęty (dla komunikatu: nie nadawaj — untalk; bądź nie odbieraj — unlisten), więc do dyspozycji pozostaje 31 różnych adresów urządzeń (włączając adres 0). O tym, czy urządzenie o wybranym adresie będzie pracowało jako nadawca czy

odbiorca, decydują bity szósty i siódmy. Jeśli bit 6 = "1", to urządzenie będzie pełniło funkcję odbiorcy, a jeśli bit 7 = "1" (bit 6 = "0") — to funkcję nadawcy.

Urządzenie, któremu użytkownik ustawił adres 5, otrzymuje poprzez magistralę danych adres "wewnętrzny" 37 i będzie wówczas pracować jako odbiorca, a jeśli ma być nadawcą, to otrzyma adres 69.

Nr urządzenia	Adresowanie odbiorców		Adresowanie nadawców	
	stan na magistrali danych	odpowiednik dziesiętny	stan na magistrali danych	odpowiednik dziesiętny
0	001 00000	32	010 00000	64
1	001 00001	33	010 00001	65
2	001 00010	34	010 00010	66
3	001 00011	35	010 00011	67
4	001 00100	36	010 00100	68
5	001 00101	37	010 00101	69
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
30	001 11110	62	010 11110	94
31	001 11111	63	010 11111	95

Na początku procedury kontroler wysyła sygnał REN, a następnie sygnał ATN ("0" lub "1", zależnie od tego, czy chodzi o dane czy też o instrukcje). Dalsze przebiegi czasowe sygnałów współpracy są pokazane na rys. 1.3.



Rys. 1.3. Diagram czasowy sygnałów przy wymianie informacji za pomocą magistrali IEC

W pierwszej chwili dane nie są jeszcze ważne (DAV = "0" — stan H) i nie wszyscy odbiorcy są przygotowani do odbioru (NRFD = "1" — stan L). Nadawca umieszcza pierwszy bajt na magistrali danych, a odbiorcy zostają po kolei przygotowani do odbioru. Kiedy wreszcie ostatni z nich zwolni swoją linię NRFD, linia NRFD systemu przechodzi w stan wysoki (H), co odpowiada zeru logicznemu. Bezpośrednio po tym nadawca, za pomocą linii DAV, zgłasza, że dane są aktualne (dane ważne). Z chwilą, gdy dane odbierze najszybciej działające urządzenie, wymusza ono na linii NRFD stan niski, czyli logiczną "1" oraz potwierdza odbiór, zwalniając swoją linię NDAC. Ponieważ jednak wszystkie wyjścia NDAC są połączone równolegle (AND), logiczne "0" (H) pojawi się na tej linii dopiero wówczas, gdy wszystkie urządzenia zakończą odbieranie danych.

Teraz nadawca ustawiając DAV = "0" sygnalizuje, że dane na magistrali są już nieważne. Urządzenia odbierające potwierdzają przyjęcie tego sygnału ustawiając "1" na linii NDAC i kolejno "zwalniają" wyjścia NRFD. W ten sposób kończy się przesłanie jednego bajtu danych.

Instrukcjom, które mogą być przesyłane w trybie instrukcji nie poświęcimy tu wiele miejsca, ponieważ są one, oraz sposób ich kodowania, podawane w opisach poszczególnych urządzeń (instrukcje obsługi). Tam też są dokładnie opisane rozkazy sterujące magistralę IEC. Z tego względu oprogramowanie tej magistrali zostanie tu potraktowane skrótowo.

W większości urządzeń istnieje możliwość sterowania wprowadzaniem lub wyprowadzaniem informacji za pomocą jednej instrukcji, na przykład:

PRINT@5: "ABC" oznacza: wysłanie informacji do urządzenia nr 5 (adres odbiorcy)

INPUT@5: A$ oznacza: odbiór informacji z urządzenia nr 5 (adres nadawcy)

Niektóre komputery traktują urządzenia zewnętrzne jako plik. Taki plik musi najpierw zostać otwarty:

OPEN 1,5 (plik 1 zostaje przypisany do urządzenia nr 5)

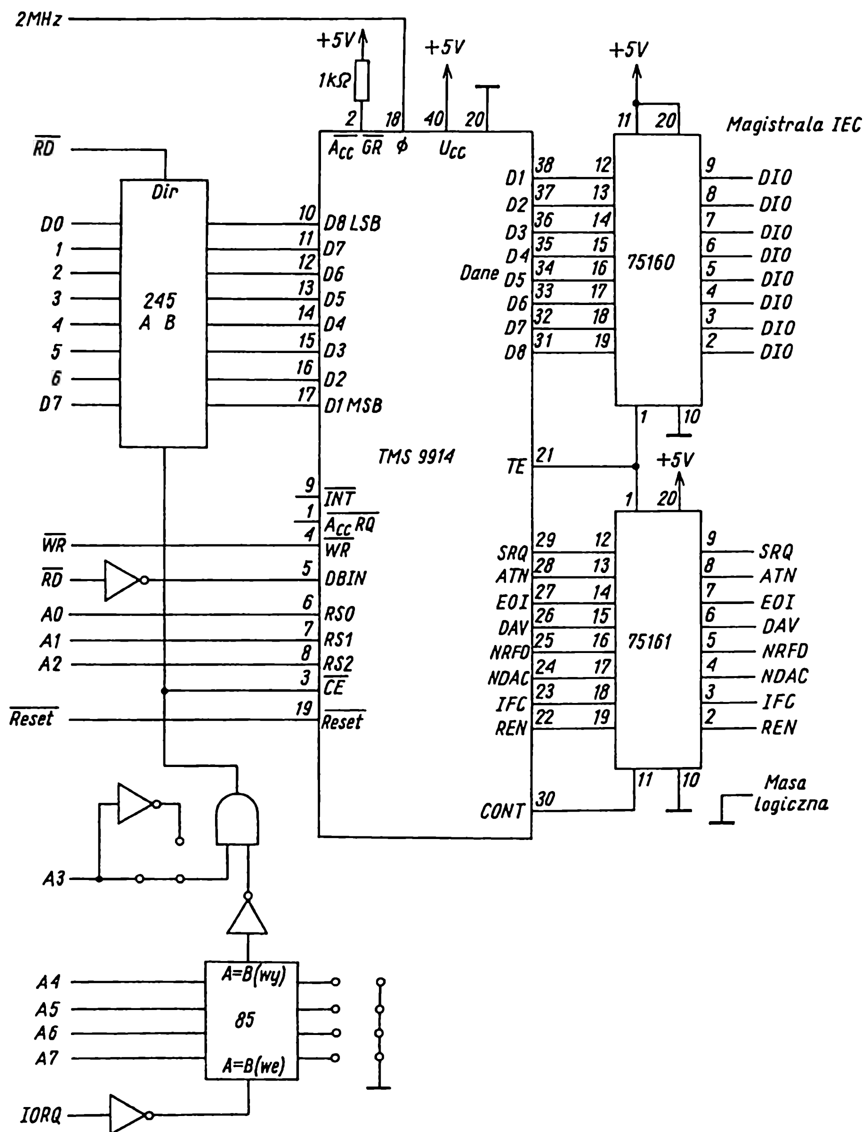
Następnie, przed każdym wprowadzeniem albo wysłaniem danych stosuje się rozkazy, np.:

PRINT # 1, "ABC" lub:

INPUT # 1, A$

Układy scalone przeznaczone do współpracy z magistralą IEC

Aby ułatwić zadanie tym czytelnikom, którzy chcieliby wyposażyć swoje urządzenia w interfejs IEC, podajemy tu rozwiązania układowe, w których



Rys. 1.4. Interfejs IEC-625 z układem TMS 9914 firmy Texas Instruments



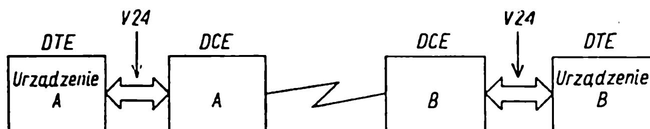
Rys. 1.5. Interfejs IEC-625 z układami 8291 i 8292 firmy Intel

znalazły zastosowanie układy scalone: kontroler 9914 firmy Texas-Instruments (rys. 1.4) oraz 8291/8292 firmy Intel (rys. 1.5).

Kontroler produkowany przez firmę Texas-Instruments nadaje się w zupełności do zastosowania w systemach z mikroprocesorem Z80.

1.1.2. Interfejs V24

W przeciwieństwie do systemu interfejsu IEC, interfejs V24 (w USA — RS 232C) nie jest systemem, który umożliwia jednoczesne dołączenie wielu urządzeń. Służy on do połączenia ze sobą dwóch urządzeń, dokładniej mówiąc — jednego urządzenia peryferyjnego (DTE — ang. *Data Terminal Equipment)*, jak komputer, drukarka itp., z urządzeniem do transmisji danych (DCE — ang. *Data Communication Equipment*) zwanym także modemem (MOdulator-DEModulator). Odległość między tymi dwoma urządzeniami nie powinna przekraczać 15 m.

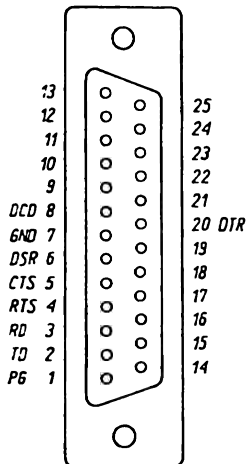


Rys. 1.6. Struktura systemu transmisyjnego z interfejsem V24

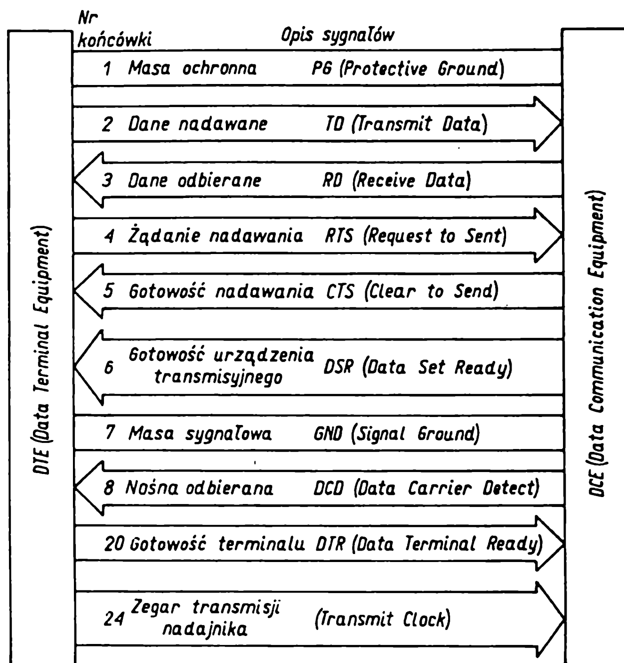
Wymiana danych odbywa się z zasady dwukierunkowo, przy czym są one przesyłane szeregowo, tzn. bit po bicie. Na rysunku 1.6 jest przedstawiona struktura toru przesyłowego (np. łącze telekomunikacyjne) między dwoma urządzeniami końcowymi, w którym został użyty interfejs V24.

Linie interfejsu i ich funkcje

Na rysunku 1.7 pokazano znormalizowane okablowanie złącza 25-stykowego, a na rys.1.8 — linie interfejsu, przewidziane normą DIN 66020. Wszystkie te linie są wykorzystywane jedynie przez niewielką liczbę użytkowników (wyprowadzenie od *1* do *8* oraz *20*). W większości przypadków wystarczają zaledwie dwa



Rys. 1.7. Znormalizowane okablowanie 25-stykowego złącza interfejsu V24



Rys. 1.8. Linie interfejsu V24 (zestaw; wg normy DIN 66020)

przewody sygnałowe i masa (wyprowadzenia *1, 2, 3*). Jeśli chce się zastosować urządzenie wyposażone w interfejs V24, to należy wpierw zapoznać się z jego instrukcją obsługi, gdzie podane jest, jakie linie interfejsu są do dyspozycji bądź jakie sygnały znajdą zastosowanie.

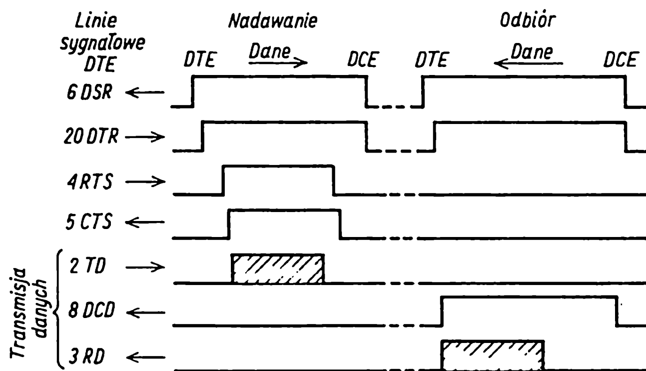
Sygnały interfejsu przedstawiono na rys. 1.8:

D1 (TD) — na tę linię urządzenie DTE wysyła dane.
D2 (RD) — z tej linii DTE odbiera dane wysyłane przez DCE (modem).
S2 (RTS) — za pomocą tej linii DTE wzywa DCE do przesłania danych.
M2 (CTS) — za pomocą tej linii DCE zgłasza do DTE swoją gotowość do przesyłania danych.
M1 (DSR) — aktywny stan tej linii wskazuje, że DCE jest gotowe do pracy, tzn. nie zaistniał żaden defekt.
M5 (DCD) — za pomocą tej linii DCE przesyła do DTE informację, że na jego (DCE) wejście jest przyłożony sygnał, którego poziom znajduje się wewnątrz ustalonego zakresu tolerancji.
T2 (TC) — po tej linii DTE przesyła do DCE częstotliwość zegarową nadawania.
T4 (RC) — ta linia służy do przesłania do DCE częstotliwości zegarowej odbioru.

S1 (DTR) — za pomocą tej linii DTE zgłasza do DCE gotowość do pracy.
T1 (TCout) — na tę linię DTE wysyła takt transmisji.

Przebieg współpracy z V24

Na rysunku 1.9 pokazano przykładowy przebieg sygnałów współpracy w interfejsie V24, prz czym terminal pracuje raz jako nadawca, a raz jako odbiorca. Strzałki oznaczają, że chodzi o sygnał wejściowy (wprowadzany do DTE) lub o sygnał wyjściowy (wychodzący z DTE).



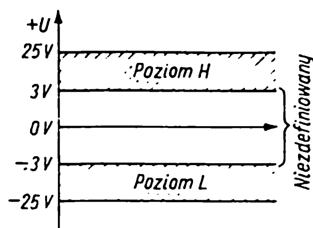
Rys. 1.9. Diagram czasowy sygnałów dla interfejsu V24

Parametry elektryczne

Na rysunku 1.10 pokazano stosowane w V24 przedziały napięć przyporządkowane poziomom: wysokiemu (H) i niskiemu (L). Maksymalna dopuszczalna bezwzględna wartość napięcia wynosi 25 V. W tablicy podano przyporządkowanie stanu L i H wartościom logicznym sygnałów:

	H	L
Sygnały współpracy i sterowania	aktywny (stan włączenia)	nieaktywny (stan wyłączenia)
Linie przesyłania danych RD, TD	0 (ang. *mark*)	1 (ang. *space*)
	bit startu	bit stopu

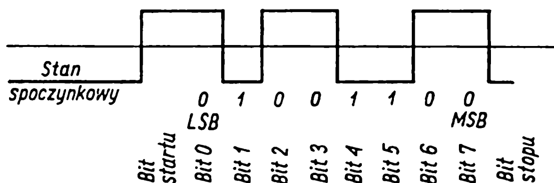
Jak wynika z tablicy, sygnały sterowania i współpracy są aktywne w stanie wysokim, a sygnały danych — w stanie niskim. Bitowi startu na linii danych odpowiada stan wysoki, a bitowi stopu — stan niski.



Rys. 1.10. Przedziały napięcia dla stanu wysokiego (H) i niskiego (L) na wejściach interfejsu V24

Przesyłanie i format danych

Jak już wspomniano, przesyłanie danych odbywa się w przypadku interfejsu V24 szeregowo — bit po bicie. Bity są przesyłane kolejno według ich wartości, tzn. najpierw bit 0, na końcu bit 7. Bit 0 jest poprzedzony bitem startu, a po ostatnim bicie danych jest przesyłany jeden lub dwa bity stopu (rys. 1.11). Kodowanie danych, które mają być przesyłane, nie jest znormalizowane. Najczęściej jest stosowany kod 7-bitowy ISO-7 albo ASCII, przy czym ósmy bit (bit nr 7) służy na ogół do kontroli parzystości.

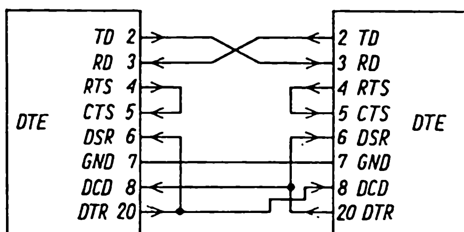


Rys. 1.11. Czasowy przebieg sygnału na linii TD/RD (przy przesyłaniu "2" = 32 w kodzie ASCII, bit 7 jest bitem parzystości)

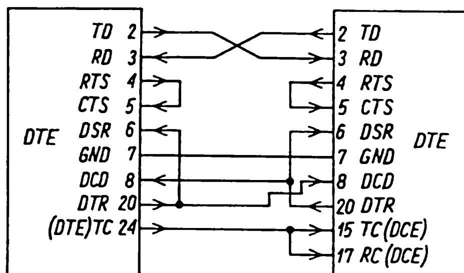
Szybkość transmisji można wybrać dowolnie spośród ustalonych wartości (ang. *baudrate*), przy czym maksymalna z nich wynosi dla interfejsu V24, zgodnie z normą DIN, niecałe 20 kbitów na sekundę (20000 bodów). Aby nie doszło do błędnej interpretacji danych, przed rozpoczęciem transmisji w obydwu urządzeniach muszą być ustalone jednakowe szybkości transmisji, o ile mają one niezależne generatory taktujące. Stosuje się następujące szybkości transmisji: 75, 110, 135, 150, 300, 600, 1200, 2400, 4800, 9600, 19200 bodów.

Bezpośrednie połączenie dwóch urządzeń końcowych

Przy odległościach kilku metrów nie ma sensu używać urządzeń do transmisji danych (modemów), wobec tego dwa współpracujące ze sobą urządzenia końcowe (np. komputer i drukarka) muszą być ze sobą połączone bezpośrednio. O ile w przypadku interfejsu V24 zaciski wyjściowe DTE musiały być połączone z odpowiadającymi im zaciskami wejściowymi DCE, to w tym przypadku występują pewne zmiany: np. wyprowadzenia *2* i *3* muszą być ze sobą skrzyżowane (inne przyporządkowane sobie pary to: *4* i *5* oraz *6* i *20*).



Rys. 1.12. Połączenie dwóch urządzeń końcowych



Rys. 1.13. Połączenie dwóch urządzeń końcowych z dodatkową linią do przesyłania sygnału taktującego

Na rysunku 1.12 pokazano, jak można połączyć ze sobą dwa urządzenia DTE, jeżeli każde z nich posiada własny generator taktujący (linie *4* i *5* też można skrzyżować między urządzeniami). Na rysunku 1.13 współpracuje taka sama para urządzeń, ale jedno z nich wytwarza sygnał taktujący dla obydwu (sprawdzić w instrukcji obsługi, czy dostępne są wejścia *15* i *17*). W przypadku przyłączenia do komputera drukarki można jego wyjście *2* połączyć z wejściem danych drukarki, a wejście *5* (linia współpracy) — z jej wyjściem BUSY (zajęty).

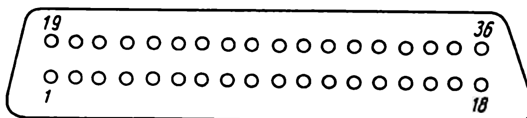
1.1.3. Interfejs równoległy — Centronics

Interfejs opracowany przez firmę Centronics, do sterowania drukarek, służy do równoległego przesyłania danych. Maksymalna odległość między nadawcą a odbiorcą wynosi 8 m, ponieważ pojemności przewodów prowadzą do sprzężeń i zniekształceń sygnałów. Szybkość transmisji zależy od możliwości sprzętu (ang. *hardware*) i teoretycznie może wynosić 1 Mbajt/s, ale wówczas długość przewodów nie może przekraczać 1 m.

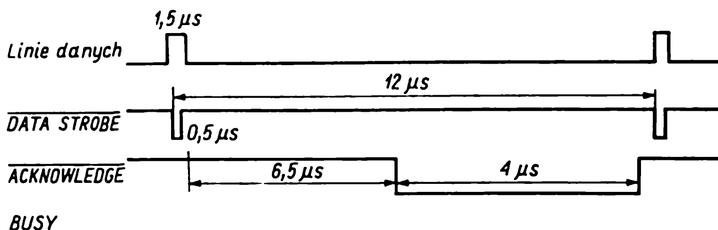
Na rysunku 1.14 pokazano schemat okablowania złącza 36-stykowego (złącze firmy Amphenol — Seria 57), a na rys. 1.15 — diagram czasowy sygnałów współpracy podczas transmisji danych.

Przesłanie jednego bajtu danych (ewentualnie 7 bitów) odbywa się w następujący sposób: urządzenie wysyłające (np. komputer) umieszcza dane (7 lub 8 bitów) na magistrali danych i po co najmniej 500 ns, wysyła sygnał strobujący o czasie trwania nie mniejszym niż 500 ns. Urządzenie odbierające dane potwierdza

1	DATA STROBE	(WE)	19	Masa – skrętka z linią	1
2	DATA 1	(WE)	20	"	2
3	DATA 2	(WE)	21	"	3
4	DATA 3	(WE)	22	"	4
5	DATA 4	(WE)	23	"	5
6	DATA 5	(WE)	24	"	6
7	DATA6	(WE)	25	"	7
8	DATA 7	(WE)	26	"	8
9	DATA 8	(WE)	27	"	9
10	ACKNLG	(WY)	28	"	10
11	BUSY	(WY)	29	"	11
12	PE(koniec papieru)	(WY)			
13	SLCT(wybór drukarki)	(WY)			
16	0V				
17	Masa obudowy				
18	+5V(z drukarki)				

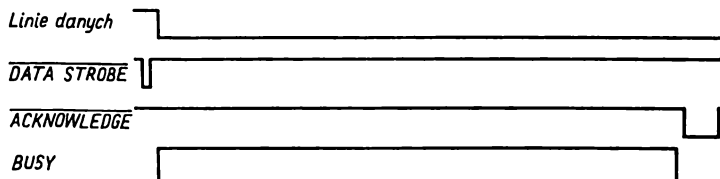


Rys. 1.14. Okablowanie złącza w standardzie Centronics



Rys. 1.15. Diagram czasowy sygnałów współpracy przy przesyłaniu informacji

ich przyjęcie po 6,5 μs sygnałem akceptacji (ang. *acknowledge)*. Jeśli bufor wejściowy zapełnił się całkowicie, np. przyjęta została liczba znaków odpowiadająca całej linii lub pojawił się kod CR (ang. *Carriage Return* — powrót karetki), to drukarka wysyła sygnał BUSY (zajęty) jako informację, że wszystkie znaki, które mogą być wydrukowane, zostały przyjęte (rys. 1.16). Po wydrukowaniu



Rys. 1.16. Diagram czasowy podczas przetwarzania (np. drukowania) przesyłanej informacji

ostatniego znaku wyjście BUSY drukarki przyjmuje z powrotem stan L, ostatnio przesłany znak zostaje potwierdzony sygnałem ACKNOWLEDGE i może rozpocząć się transmisja nowych znaków.

W wielu drukarkach przesłanie danych odbywa się podczas powrotu karetki. Jeśli przyrząd odbierający dane nie ma bufora wejściowego dla większej liczby znaków, powinien natychmiast po przyjęciu znaku przystąpić do jego przetwarzania i wysyłać sygnał BUSY tak długo, jak długo przetwarzanie nie zostanie zakończone. Dopiero potem następuje potwierdzenie przyjęcia ostatniego znaku sygnałem ACKNOWLEDGE (rys. 1.16). Odnośnie parametrów elektrycznych: poziomy sygnałów wyjściowych i wejściowych interfejsu są kompatybilne z poziomami sygnałów TTL, co znacznie upraszcza jego konstrukcję.

Jeżeli trzeba użyć interfejsu Centronics w celu sterowania, wystarczy w języku Basic wysyłać dane za pomocą instrukcji LPRINT$.

1.2. Interfejs specjalny

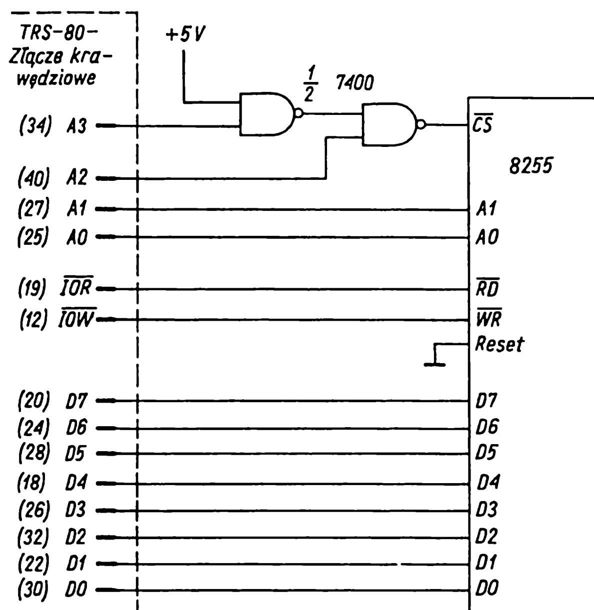
Jak już wspomniano, układy zaproponowane w tej książce zostały zaprojektowane z myślą o komputerze TRS-80 firmy Tandy (Radio Shack). Komputer ten nie ma w swoim wyposażeniu podstawowym żadnego z opisanych wyżej interfejsów standardowych. Jeżeli dołączyć do niego zalecany przez wytwórnię interfejs rozszerzający, otrzymuje się do dyspozycji interfejs w standardzie Centronics. Ten jednak wymaga stosowania sygnałów zgodnych z regułami sterowania drukarki. Zaprojektowano więc układ interfejsu dołączony wprost do złącza bezpośredniego, które znajduje się po lewej stronie obudowy klawiatury. Zdarza się jednak, że trzeba równocześnie dołączyć wymieniony interfejs drukarki, co wymaga dostępu do tego właśnie złącza. Z tego względu, w lewej przedniej części interfejsu fabrycznego jest umieszczone drugie takie samo złącze bezpośrednie, na które podaje się te same sygnały. W obu złączach znajdują się więc wszystkie niezbędne wyprowadzenia danych oraz część magistrali adresowej (A0 do A3). Jako sygnały sterujące używane są linie IN oraz OUT.

Na rysunku 1.17 pokazano budowę interfejsu. Składa się on z programowanego układu wejścia/wyjścia 8255 firmy Intel. Układ ten posiada trzy porty[1)]: A, B i C, po osiem wyprowadzeń każdy, dzięki czemu możliwa jest równoległa transmisja informacji (tzn. bajt po bajcie) przez każdy z nich.

Oznaczenia spotykane na kartach katalogowych i schematach to najczęściej kombinacje PA, PB i PC jako skrócona forma od ang. Port A, B lub C. Skróty te mają zastosowanie również przy oznaczaniu poszczególnych wypro-

---

1) W polskiej literaturze dotyczącej techniki mikroprocesorowej nazywane też bramami (przyp. tłum.).



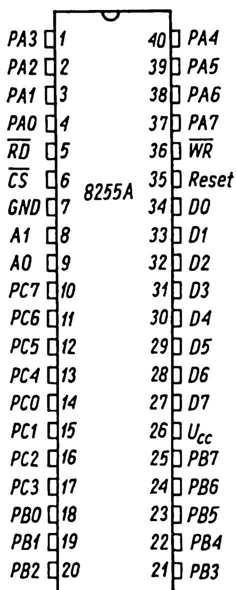
Rys. 1.17. Interfejs specjalny — schemat ideowy. Podane liczby są numerami styków złącza TRS-80

wadzeń (patrz także rys. 1.18), np. PB0 lub PB7 rozumiane jest jako wyprowadzenie o numerze 0 lub 7 portu B (w angielskojęzycznej literaturze o mikroprocesorach końcówki lub linie magistrali systemowej oraz portów we/wy oznaczane są odpowiednio do ich wagi, a więc pierwsza pozycja jest oznaczana indeksem 0, np. PC0, a pozycja najbardziej znacząca (pozycja odpow. 128) ma indeks 7).

Każdy z portów 8255 może pracować jako wejściowy lub wyjściowy. Jego funkcja jest każdorazowo ustalana za pomocą słowa sterującego wprowadzanego do logiki sterującej układu poprzez magistralę danych.

Na rysunku 1.18 pokazano układ wyprowadzeń podstawki 8255. Układ staje się aktywny w momencie wybrania go za pomocą linii adresowych *2* i *3*, i przesłanie sygnałów IN (operacja odczytu) lub OUT (operacja zapisu do elementu) przez TRS-80 (takie linie sygnałowe dla odczytu albo zapisu istnieją w takiej samej lub podobnej postaci w większości komputerów). Linie A0 i A1 magistrali adresowej decydują, do którego portu odnosi się operacja zapisu — WY bądź odczytu — WE (patrz tablica poniżej).

A1	A0	Przepływ informacji
0	0	linie danych ⟷ port A
0	1	„ ⟷ port B
1	0	„ ⟷ port C
1	1	„ ⟶ rejestr sterowania



Opis wyprowadzeń

D_0–D_7	Magistrala danych (dwukierunkowa)
RESET	Zerowanie (WE)
$\overline{CS}$	Wybór elementu
$\overline{RD}$	Odczyt (WE)
$\overline{WR}$	Zapis (WE)
A0, A1	Adres portu
PA_0–PA_7	Port A (bity 0–7)
PB_0–PB_7	Port B (bity 0–7)
PC_0–PC_7	Port C (bity 0–7)
U_{cc}	Zasilanie (+5 V)
GND	Masa (0 V)

Rys. 1.18. Układ i opis wyprowadzeń układu 8255

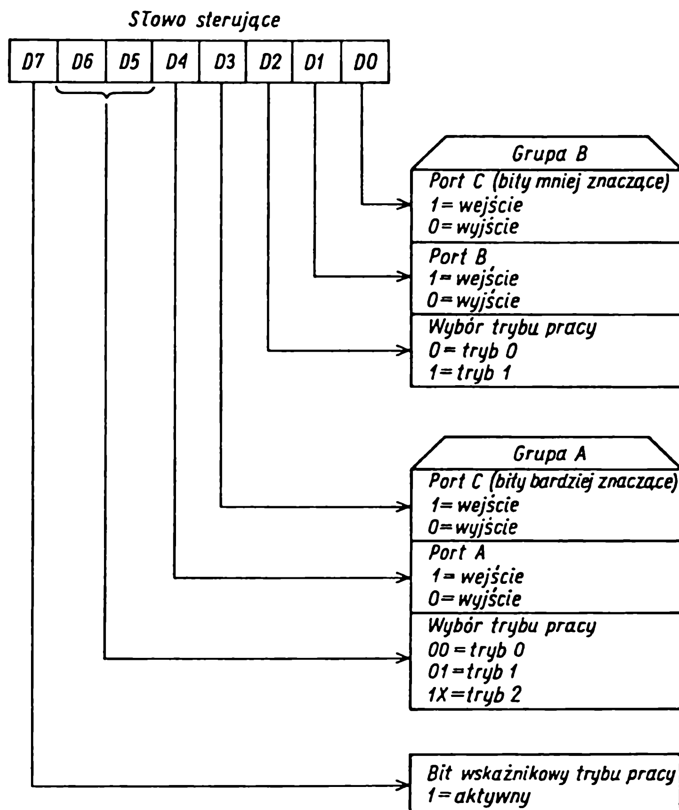
Zaprogramowanie układu 8255 realizuje się za pomocą słowa sterującego wysyłanego przez komputer. Sposób tworzenia słowa sterującego wynika z rys. 1.19. Załóżmy na przykład następujące przyporządkowanie funkcji:

Port A Wyjście
Port B Wejście
Port C Wejście

Dla tego przypadku otrzymuje się następujące binarne słowo sterujące $10001011_2 = 139_{10}$ (bit oznaczający tryb pracy ma być przy programowaniu stale równy 1!). Wybrany został tryb (rodzaj) pracy "0", który będzie później stale wykorzystywany. W zasadzie wyprowadzenie informacji za pomocą programu w języku Basic odbywa się przy użyciu instrukcji: OUT *adres portu*, *dana* (*dana* — słowo 8-bitowe, może być podane w postaci dziesiętnej albo jako zmienna, której wartość została ustalona wcześniej w programie), a wprowadzenie za pomocą instrukcji: INP (*nr portu*).

Porty mają następujące adresy:

Port	Adres
A	4
B	5
C	6
Rejestr sterowania	7



Rys. 1.19. Format słowa sterującego dla układu 8255

Stąd też, aby zaprogramować funkcje portów w układzie wejścia/wyjścia zgodnie z powyższymi założeniami, należy umieścić w programie instrukcję: OUT 7,139.

Odczyt wartości zmiennej poprzez port A interfejsu wyglądałby następująco: X = INP(4). Wyżej wspomniane zaprogramowanie układu interfejsu musi być zrealizowane, zanim pierwsze słowo danych zostanie wprowadzone do portów A, B lub C.

2. Pomiary za pomocą komputera wyposażonego w Basic

Należy przede wszystkim pozbyć się złudzenia co do tego, że zastosowanie komputera uczyni zbędnymi zakup i użycie kosztownych przyrządów pomiarowych. Jego zadaniem powinno być jedynie cyfrowe przetworzenie danych pomiarowych dostarczanych przez zewnętrzne bloki pomiarowe i wizualizacja wyników na ekranie lub przedstawienie ich w postaci wydruku, w formacie narzuconym przez użytkownika.

2.1. Pomiary wielkości wolnozmiennych

Wielkości wolnozmienne są to takie wielkości, których szybkość zmian (mała w porównaniu z szybkością przetwarzania komputera wyposażonego w interpreter Basicu) wymaga pobierania i przetwarzania próbek ich wartości nie częściej niż od 1 do 3 razy na sekundę.

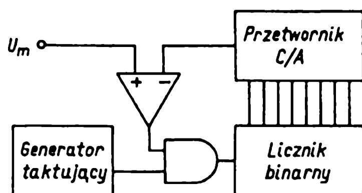
2.1.1. Pomiar napięcia

Parametry

- pomiar napięć dodatnich w zakresie od 0 do 2,5 V z podziałem na 256 poziomów (przetwornik 8-bitowy),
- czas pomiaru ok. 1 ms,
- możliwość powiększenia zakresu pomiarowego za pomocą dzielnika napięcia,
- nieliniowość ± 1/2 LSB.

Sprzęt (układ pomiarowy)

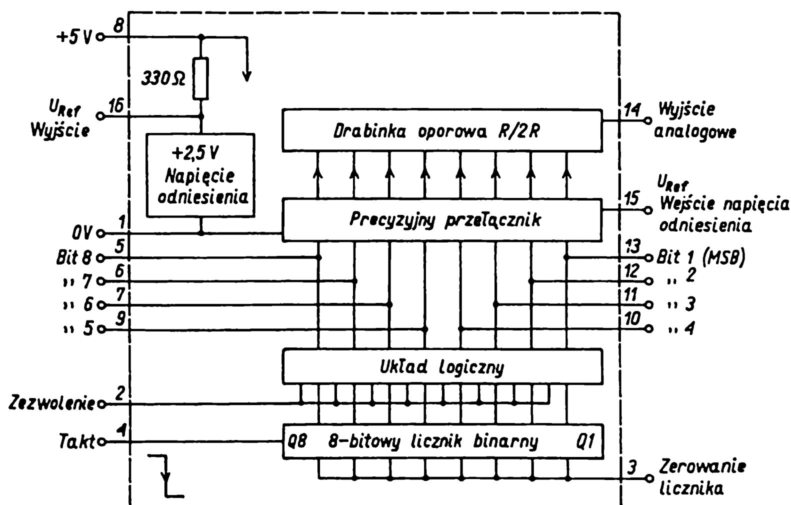
Pomiar napięcia jest realizowany przez przetwornik analogowo-cyfrowy A/C (ang. *Analog to Digital Converter*) z kompensacją liniową, przy czym w układzie



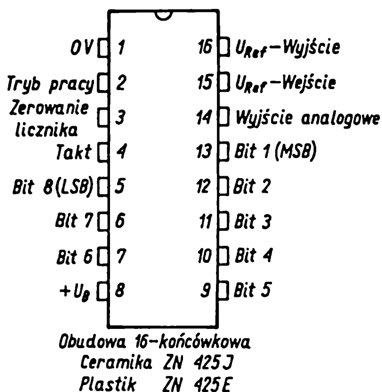
Rys. 2.1. Metoda kompensacji liniowej — zasada działania

zastosowano 8-bitowy przetwornik ZN 425 firmy Ferranti. Rysunek 2.1 ilustruje zasadę działania tego przetwornika. Impulsy z generatora taktującego zliczane są przez licznik binarny, do którego wyjść jest dołączona drabinka oporowa przetwornika cyfrowo-analogowego. Dzięki temu na wyjściu drabinki pojawia się napięcie, którego wartość chwilowa jest proporcjonalna do stanu licznika — napięcie schodkowe. Jest ono porównywane za pomocą komparatora (wzmacniacza operacyjnego) z napięciem mierzonym. W momencie, w którym wartość napięcia schodkowego przekroczy wartość napięcia mierzonego, komparator wymusi na jednym z wejść bramki AND stan niski (L), co spowoduje zatrzymanie procesu liczenia. Stan licznika może być teraz uznany za cyfrową reprezentację mierzonego napięcia.

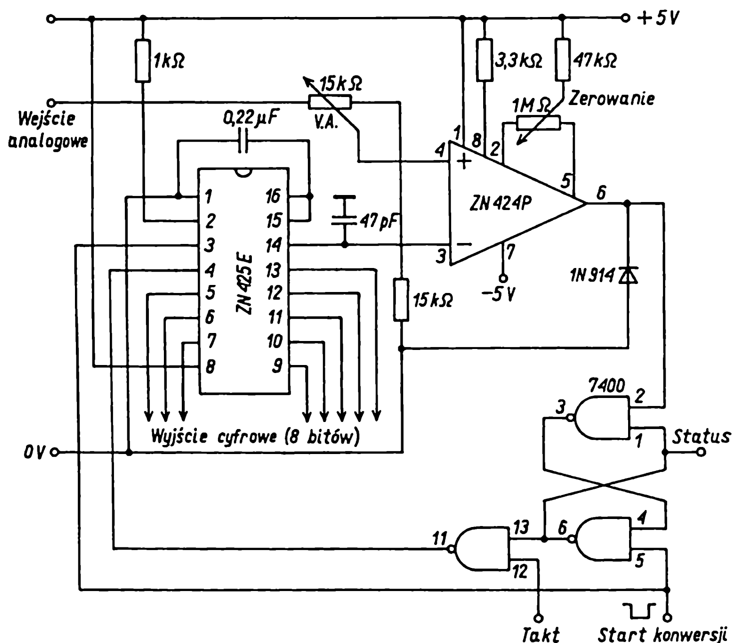
Na rysunku 2.2 pokazano schemat blokowy przetwornika ZN 425, a na rys. 2.3 — układ jego wyprowadzeń. Pełny układ przetwornika (propozycja wykorzystania układu ZN 425 zalecana przez firmę Ferranti) przedstawiono na rys. 2.4, a na rys. 2.5 podano schemat generatora taktującego. Zamiast wzmacniacza operacyjnego ZN 424P (rys. 2.4) można użyć bardziej popularnego



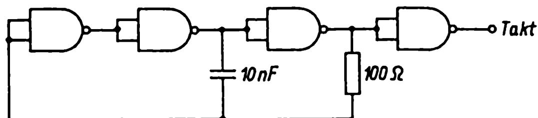
Rys. 2.2. Schemat blokowy układu ZN 425



Rys. 2.3. Wyprowadzenia układu ZN 425 (widok z góry).
Uwaga: Bit 8 jest bitem najmniej znaczącym, a bit 1 najbardziej znaczącym



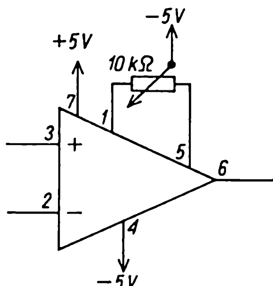
Rys. 2.4. Schemat ideowy przetwornika analogowo-cyfrowego



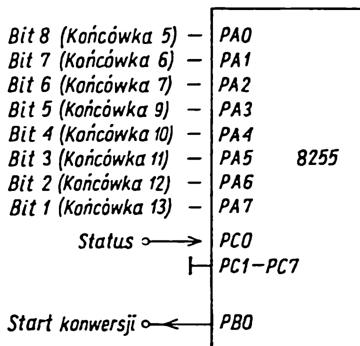
Rys. 2.5. Generator taktujący dla układu przetwornika A/C (US = 7400)

układu µA 741 (rys. 2.6). W tym przypadku mogą się jednak pojawić trudności z ustawieniem zera (kompensacja napięcia niezrównoważenia) ze względu na jego duże wzmocnienie.

Czas przetwarzania opisywanego przetwornika wynosi ok. 1 ms. Ponieważ pełny cykl zliczania wymaga 256 taktów (licznik 8-bitowy), więc maksymalna częstotliwość generatora taktującego wynosi 256 kHz. Na rysunku 2.7 pokazano sposób połączenia przetwornika z interfejsem komputera.



Rys. 2.6. Przyłączenie układu µA 741

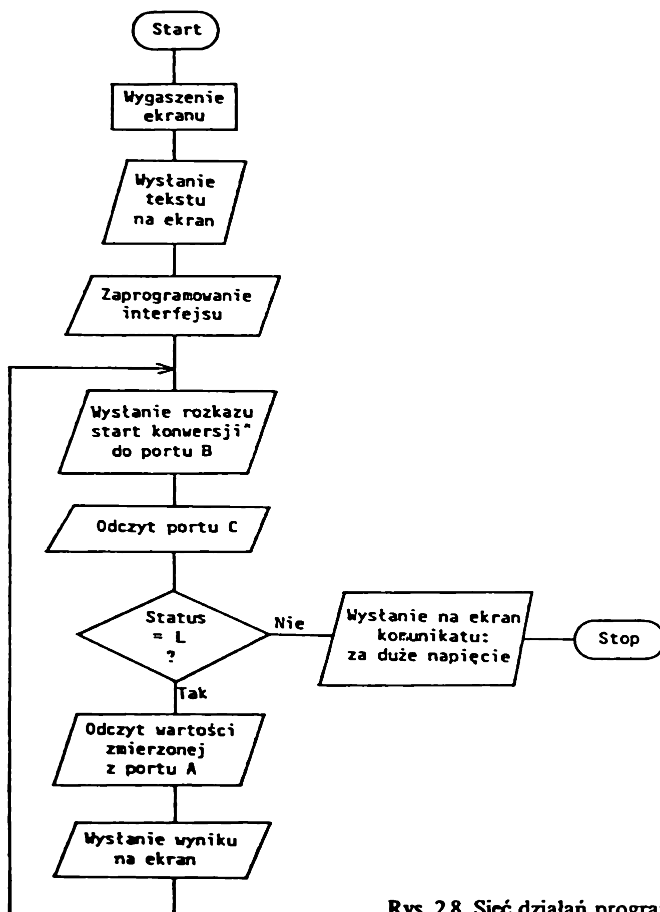


Rys. 2.7. Przyłączenie układu przetwornika do interfejsu komputera

Przebieg procesu konwersji: Impuls "start konwersji" zeruje 8-bitowy licznik binarny oraz ustawia wysoki stan na wyjściu przerzutnika R-S (wyjście "status") zbudowanego z dwóch bramek NAND. Stan wysoki (H) na wyjściu "status" sygnalizuje, że odbywa się przetwarzanie. W momencie, gdy zawartość licznika osiągnie stan odpowiadający mierzonemu napięciu, komparator (ZN 424P) wymusi zmianę stanu przerzutnika R-S. Stan niski (L) na wyjściu przerzutnika (wyjście "status" przetwornika) stanowi informację dla komputera, że przemiana się zakończyła i dane mogą być odczytane.

Program

Na rysunku 2.8 przedstawiono sieć działań programu "POMIAR NAPIĘCIA". Ponieważ w użytym przetworniku A/C nie ma wskaźnika przepełnienia, więc kontrola przepełnienia musi być zrealizowana programowo. Przekroczenie zakresu pomiarowego objawia się w ten sposób, że po osiągnięciu pełnego zakresu licznik zaczyna zliczać od początku, a wskaźnik "status" pozostaje w stanie wysokim (H). Dlatego też odczyt stanu licznika (wartość napięcia w postaci cyfrowej) następuje dopiero po sprawdzeniu stanu wyjścia "status". Jeśli wyjście "status" nie jest w stanie L, to komputer wyświetli na ekranie komunikat "NAPIĘCIE ZA WYSOKIE". W przeciwnym przypadku zmierzona wartość



Rys. 2.8. Sieć działań programu "POMIAR NAPIĘCIA"

cyfrowa zostaje pomnożona przez wartość napięcia odpowiadającą jednemu "schodkowi" (0,01 V) i wysłana na ekran.

Na rysunku 2.9 pokazano program napisany w języku Basic. Pierwsza część tego programu (linie 130—190) dokonuje przygotowania ekranu, w linii 210 następuje zaprogramowanie interfejsu, a linia 220 rozpoczyna procedurę pomiarową, tzn. odczyt, przetwarzanie oraz wizualizację zmierzonej i cyfrowo przetworzonej wartości napięcia.

Regulacja układu

Po zmontowaniu układu, dołączeniu go do interfejsu komputera i wystartowaniu programu, przystępujemy do regulacji. Na początku należy zewrzeć do

```
100 REM **** PROGRAM : "POMIAR NAPIECIA"              ****
110 REM **** NAZWA PROG.: "POMNAP"                    ****
120 CLS
130 REM LINIE 140-190 : RAMKA NA EKRAN
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) SET (127,Y)
190 NEXT Y
200 PRINT@ 328;"P O M I A R   N A P I E C I A"
210 OUT 7,153 : REM ZAPROGRAMOWANIE INTERFEJSU
220 OUT 5,0 : OUT 5,1
230 STAT = INP (6)
240 IF STAT > 0 PRINT@ 584,
250 X = INP (4)
260 U = X * 0.01 : REM WSPOLCZYNNIK 10 MILIWOLTOW NA KWANT
270 PRINT @ 648,"NAPIECIE WYNOSI :"
280 M$ ="#####.##"
290 PRINT @712,USING M$;U;
300 PRINT " V"
310 FOR I = 1 TO 100 : NEXT I  : REM  PETLA OPOZNIENIA
320 GOTO 220
```

Rys. 2.9. Program "POMIAR NAPIĘCIA"

masy wejście analogowe i potencjometrem "zerowanie" sprowadzić wskazanie do zera. Następnie, używając źródła napięcia o znanej wartości, ustawić właściwe wskazanie za pomocą potencjometru VA. Maksymalna wartość napięcia pomocniczego nie powinna przekraczać 2,55 V. W układzie ZN 425 pojedynczy krok kwantowania (jeden "schodek") wynosi dokładnie 10 mV. Jeśli wymagany jest większy zakres pomiarowy, to łatwo obliczyć, o jaką wartość należy zwiększyć rezystancję potencjometru VA. W tym przypadku trzeba odpowiednio skorygować 260 linię programu dostosowując mnożnik do nowego kroku kwantowania.

2.1.2. Pomiar prądu

Pomiar prądu nie przedstawia specjalnych trudności. Przeprowadzany jest zwykle za pomocą multimetru jako pomiar spadku napięcia na rezystancji wtrąconej w obwód prądu. Rezystancja ta powinna być możliwie mała, aby nie obciążała zbytnio obwodu i nie wprowadzała błędu. Przeliczenie zmierzonego napięcia na odpowiadającą mu wartość prądu jest zrealizowane programowo np. przy zastosowaniu instrukcji:

```
100 I = U/R
```

przy czym R — wartość rezystancji pomiarowej.

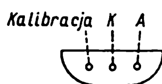
2.1.3. Pomiar temperatury

Parametry

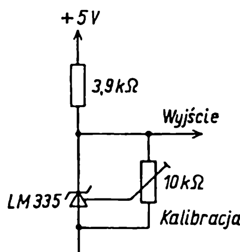
- zakres pomiaru temperatury od 0 do 100°C,
- możliwość programowego rozszerzenia zakresu pomiarowego na temperatury ujemne do −10°C.

Sprzęt (układ pomiarowy)

Pomiar temperatury umożliwia przetwornik, który przetwarza temperaturę na proporcjonalne do niej napięcie. Stosowany jest czujnik temperatury LM 335, którego charakterystyka przetwarzania ma nachylenie 10 mV/K, a napięcie wyjściowe w temperaturze pokojowej (20°C) wynosi 2,73 V. Zakres pomiarowy LM 335 obejmuje wartości temperatury od −10°C do +100°C. Napięcie wyjściowe czujnika temperatury jest mierzone przez układ do pomiaru napięcia (przetwornik A/C), a następnie przetwarzane w postaci cyfrowej przez komputer. Na rysunku 2.10 pokazano układ wyprowadzeń czujnika, a na rys. 2.11 — standardowy układ aplikacyjny. Równolegle dołączony potencjometr umożliwia dokładne dobranie nachylenia charakterystyki napięcie-temperatura równe 10 mV/K. Wartość rezystora szeregowego nie jest krytyczna, należy zwrócić jedynie uwagę na to, aby wartość prądu przepływającego przez układ nie przekroczyła 5 mA. Na ogół wybiera się stosunkowo dużą wartość rezystancji, aby ograniczyć nagrzewanie się czujnika wywołane przepływającym przezeń prądem.

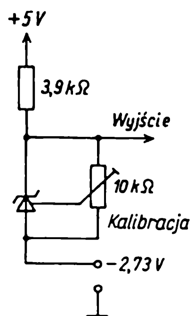


Rys. 2.10. Układ wyprowadzeń czujnika temperatury LM 335 (widok od spodu)



Rys. 2.11. Standardowy sposób przyłączenia czujnika temperatury LM 335

W większości przypadków nie jest pożądany pomiar temperatury w K (kelwinach). Wprawdzie łatwo dałoby się za pomocą programu uzyskać przeliczenie na stopnie Celsjusza odejmując od zmierzonej wartości 273, ale niestety trzeba by w zastosowanym przetworniku A/C zwiększyć krok kwantowania i zakres napięcia wejściowego. To prowadziłoby z kolei do zmniejszenia rozdzielczości wskazania, której odpowiada tylko 256 stopni kwantowania.



Rys. 2.12. LM 335 z dodatkowym źródłem napięcia

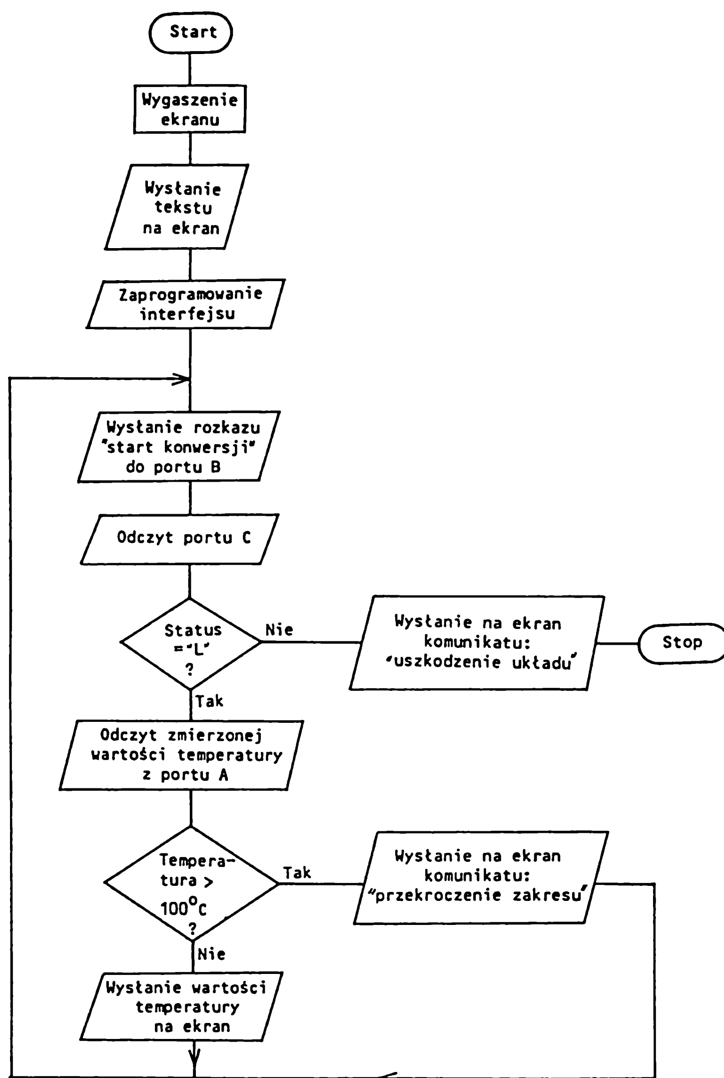
Proste rozwiązanie tego problemu polega na przesunięciu potencjału zera czujnika temperatury do −2,73 V (rys. 2.12). Użyte w tym celu napięcie pomocnicze musi być jednak bardzo stabilne, ponieważ zmiana tego napięcia o 10 mV spowoduje błąd wskazania o 1 stopień. Dlatego zaleca się stosować do tego celu scalony stabilizator napięcia. Na wyjściu stabilizatora powinien być włączony potencjometr, którym można by dokładnie ustalić wartość napięcia pomocniczego.

Program

Na rysunku 2.13 przedstawiono sieć działań programu "POMIAR TEMPERATURY". Ponieważ zakres pomiarowy czujnika obejmuje temp. od 0°C do 100°C, tzn. maksymalna wartość napięcia wynosi 1 V, nie może wystąpić przekroczenie zakresu przetwornika A/C. Jeśli jednak zdarzy się uszkodzenie układu pomiarowego, to zostanie ono zasygnalizowane użytkownikowi. Na ekranie pojawi się również ostrzeżenie, jeśli temperatura czujnika przekroczy 100°C. Na rysunku 2.14 pokazano program napisany w języku Basic. W jego pierwszej części odbywa się graficzne przygotowanie ekranu (wysłanie ramki na ekran). W linii 210 zostaje zaprogramowany interfejs komputera. Od linii 220 rozpoczyna się właściwa część pomiarowa programu (wysłanie "L" a następnie "H" na końcówkę PB0 portu B interfejsu, co odpowiada wysłaniu 0 i 1 w systemie dziesiętnym), odczytanie, a następnie wysłanie na ekran zmierzonej wartości temperatury.

Budowa i regulacja układu

Układ czujnika dołącza się do miernika napięcia z p. 2.1.1 (rys. 2.4 i 2.7). Po wprowadzeniu i wystartowaniu właściwego programu rozpoczyna się cechowanie czujnika. Zaczynamy od ustawienia punktu zerowego. W tym celu zanurza się czujnik w mieszaninie wody z lodem. Po ustaleniu się temperatury (0°C) oraz doregulowaniu wartości napięcia pomocniczego za pomocą woltomierza cyfrowego, regulujemy zero na wejściu tak, aby uzyskać wskazanie 0°C (o ile wymagane jest wskazanie w °C). Jeśli nie ma do dyspozycji woltomierza cyfrowego, ustawia się najpierw napięcie pomocnicze tak, aby otrzymać wska-



Rys. 2.13. Sieć działań programu "POMIAR TEMPERATURY"

zanie 0°C. Następnie powtarza się pomiar przy wyższej temperaturze, której wartość mierzy się możliwie dokładnym termometrem, i za pomocą potencjometru VA ustala się właściwe wskazanie komputera. Czynność tę powtarza się kilkakrotnie, ponieważ poszczególne organy regulacyjne wpływają na siebie wzajemnie.

```
100 REM **** PROGRAM :"POMIAR TEMPERATURY"   ****
110 REM **** NAZWA PROG.: "POMTEMP"          ****
120 CLS
130 REM LINIE 140-190 : RAMKA NA EKRAN
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) SET (127,Y)

190 NEXT Y
200 PRINT@ 328;"P O M I A R  T E M P E R A T U R Y";
210 OUT 7,153 : REM ZAPROGRAMOWANIE INTERFEJSU
220 OUT 5,0 : OUT 5,1
230 STAT = INP (6)
240 IF STAT > 0 PRINT@ 584, "USZKODZENIE UKLADU"; : END
250 T = INP (4)
260 IF T > 100 PRINT@ 648 , "PRZEKROCZENIE ZAKRESU"; GOTO 220
270 PRINT @ 648,"TEMPERATURA WYNOSI : ";
280 M$ = "###"
290 PRINT @712,USING M$;
300 PRINT " STOPNI CELSJUSZA";
310 FOR I = 1 TO 100 : NEXT I  : REM  PETLA OPOZNIENIA
```

Rys. 2.14. Program "POMIAR TEMPERATURY"

Rozszerzenie zakresu pomiarowego

W wielu przypadkach podany wyżej zakres pomiarowy nie wystarcza. Jeśli np. będziemy mierzyć temperaturę poniżej zera, na wyjściu czujnika pojawi się napięcie ujemne. Ponieważ przetwornik mierzy jedynie napięcia dodatnie, zagadnienie to można rozwiązać stosując pewien trik.

Zamiast pokazanego na rys. 2.12 napięcia −2,73 V, zastosujemy np. napięcie − 2,53 V. W ten sposób na wyjściu czujnika otrzymamy napięcie równe 0 V przy temperaturze −20°C. A więc temperaturze −20°C odpowiada wartość cyfrowa równa 0. W programie także trzeba od odczytanej wartości (linia 250) odjąć 20:

250 T = INP(4) − 20

Jeśli trzeba w sposób istotny rozszerzyć granice mierzonych temperatur, używa się czujnika LM 135, który obejmuje zakres temperatur od −55 do +150°C.

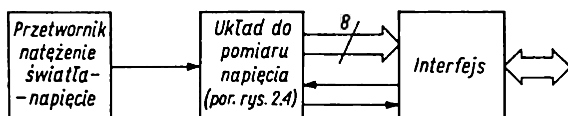
2.1.4. Pomiar natężenia oświetlenia

Parametry

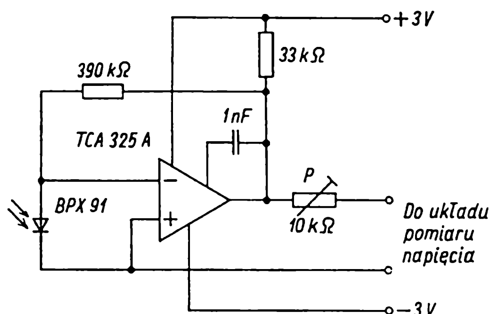
- pomiar natężenia oświetlenia do 1000 luksów,
- możliwość rozszerzenia zakresu pomiarowego do 100 000 luksów.

Sprzęt (układ pomiarowy)

Na rysunku 2.15 pokazano schemat blokowy układu do pomiaru natężenia oświetlenia. Napięcie pobrane z końcówek fotoelementu zostaje wzmocnione i podane na wejście przetwornika A/C (układ do pomiaru napięcia przedstawiono na rys. 2.4; patrz p. 2.1.1). Stąd sygnał przetworzony na postać cyfrową jest przesyłany za pośrednictwem interfejsu do komputera. Na rysunku 2.16



Rys. 2.15. Schemat blokowy urządzenia do pomiaru natężenia oświetlenia



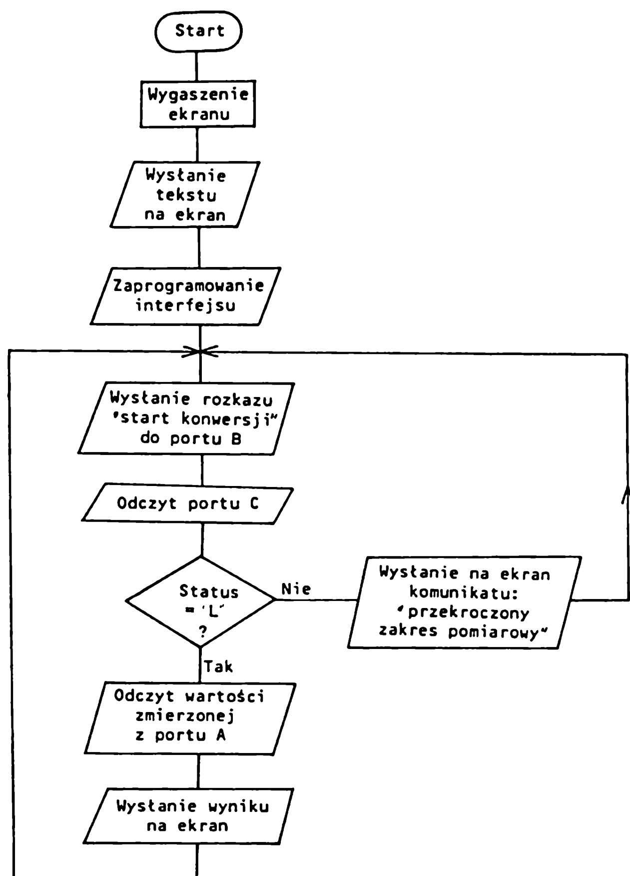
Rys. 2.16. Układ do pomiaru natężenia oświetlenia

pokazano układ pracy elementu światłoczułego. Napięcie z fotoelementu jest wzmacniane przez wzmacniacz operacyjny. Maksymalne napięcie wyjściowe zależy od wzmocnienia wzmacniacza określonego wartością rezystancji sprzężenia zwrotnego (390 kΩ).

Program

Na rysunku 2.17 pokazano sieć działań programu "POMIAR NATĘŻENIA OŚWIETLENIA". Aby zapobiec błędnemu wskazaniu, w programie sprawdzany jest stan linii statusu przetwornika. W przypadku, gdy napięcie na wejściu przetwornika jest zbyt duże, na ekranie pojawia się informacja "PRZEKROCZONY ZAKRES POMIAROWY".

Na rysunku 2.18 przedstawiono program napisany w Basicu. W pierwszej części programu następuje przygotowanie ekranu monitora. W linii 210 zostaje zaprogramowany interfejs, a od linii 220 rozpoczyna się właściwy pomiar (wysłanie "L" a następnie "H" na PB0). Wreszcie, o ile nie nastąpiło przekroczenie zakresu, następuje przetworzenie i wysłanie na ekran wartości zmierzonego natężenia oświetlenia.



Rys. 2.17. Sieć działań programu "POMIAR NATĘŻENIA OŚWIETLENIA"

Regulacja układu

Regulacji części przetwornikowej dokonuje się zgodnie z opisem przedstawionym w p. 2.1.1. Trudności mogą pojawić się podczas dokładnej regulacji części analogowej. Jeśli można wypożyczyć luksomierz, to należy ten wycechowany przyrząd umieścić w takiej samej odległości od źródła światła jak fotoelement, przy czym natężenie oświetlenia powinno sięgać 1000 luksów (pełny zakres). Następnie należy potencjometr układu pomiarowego ustawić tak, aby na ekranie także pojawiło się wskazanie 1000 luksów. Jeżeli nie ma do dyspozycji luksomierza, można skorzystać z następującego przybliżenia: natężenie oświetlenia wytwarzane przez 40-watową żarówkę w punkcie odległym od niej o ok. 18 cm wynosi 1000 Lx.

```
100 REM **** PROGRAM :"POMIAR NATEZENIA OSWIETLENIA"       ****
110 REM **** NAZWA PROG.: "NATOSW"                         ****
120 CLS
130 REM LINIE 140-190 : RAMKA NA EKRAN
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) SET (127,Y)
190 NEXT Y
200 PRINT@ 328;"POMIAR NATEZENIA OSWIETLENIA"
210 OUT 7,153 : REM ZAPROGRAMOWANIE INTERFEJSU
220 OUT 5,0 : OUT 5,1
230 STAT = INP (6)
240 IF STAT > 0 PRINT@ 584, "PRZEKROCZONY ZAKRES              ";: GOTO220
250 X = INP (4)

260 U = X * 10  : REM WSPOLCZYNNIK DLA ZAKR. 1000 LX
270 PRINT @584,"NATEZENIE OSWIETLENIA WYNOSI :";
280 M$ ="#####.##"
290 PRINT @712,USING M$;
300 PRINT " LX";
310 FOR I = 1 TO 100 : NEXT I  : REM  PETLA OPOZNIENIA
320 GOTO 220
```

Rys. 2.18. Program "POMIAR NATĘŻENIA OŚWIETLENIA"

Ulepszenia

Zakres pomiarowy przyrządu można łatwo rozszerzyć. Wymiana rezystora 390 kΩ na inny o 10- lub 100-krotnie mniejszej wartości spowoduje 10- lub 100-krotne rozszerzenie zakresu. Trzeba też odpowiednio zmienić współczynnik w linii 260 programu.

Można również rozbudować program w ten sposób, że na przykład pomiar oświetlenia będzie przeprowadzany w dłuższym okresie czasu, a wyniki pomiarów zostaną przedstawione na ekranie graficznie w postaci rozkładu czasowego natężenia światła. Jeśli kolejne pomiary zostaną przeprowadzone w jednakowych odstępach czasu, można określić wartość średnią dzieląc zsumowane wartości przez liczbę pomiarów (lub liczbę odstępów czasowych). Takie regularne odstępy czasu można łatwo zrealizować programowo, wykorzystując w tym celu programowy zegar komputera.

Urządzenia do pomiaru natężenia oświetlenia powinny obejmować dokładnie część widzialną widma. Niestety, charakterystyka widmowa czułości fotoelementu różni się od charakterystyki oka ludzkiego. Jeżeli zależy nam na dużej dokładności pomiarów, to należy użyć dodatkowo optycznych filtrów korekcyjnych (np. BG 38 firmy Glasswerke Schott & Gen., Mainz).

Wymagane wartości natężenia oświetlenia
60 Lx — przedpokój, spiżarnie
120 Lx — WC, pokój dziecinny
500 Lx — jadalnia, kuchnia, hobby, prace laboratoryjne
750 Lx — czytanie, pisanie
1000 Lx — rysunek techniczny, prace precyzyjne

2.1.5. Pomiar stanu wypełnienia zbiornika

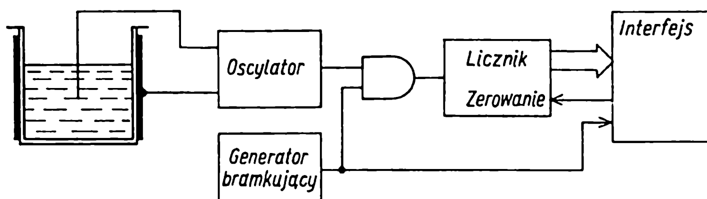
Parametry

- układ umożliwia wskazanie wysokości lustra dowolnej cieczy w zbiorniku (cztery poziomy dyskretne).

Sprzęt (układ pomiarowy)

Do pomiarów wysokości lustra cieczy w zbiorniku używa się dwóch metod: pierwsza — "elegancka" pod względem technicznym — umożliwia dowolnie dokładne wskazanie, druga — uproszczona — pozwala jedynie na wskazanie kilku dyskretnych poziomów.

Na rysunku 2.19 przedstawiono schemat blokowy rozwiązania "eleganckiego". Zewnętrzna strona zbiornika wykonanego ze szkła, ewentualnie z tworzywa sztucznego, oklejona jest w dużej części folią metalową. Folia wraz z cieczą wypełniającą naczynie tworzy kondensator, którego pojemność zależy bezpośrednio od stanu napełnienia. Ten kondensator jest elementem określającym częstotliwość generatora fali prostokątnej. Impulsy z generatora są



Rys. 2.19. Schemat wyjaśniający zasadę ciągłego pomiaru stanu wypełnienia zbiornika

zliczane w ciągu odpowiednio dobranego czasu (czas bramkowania) przez licznik, którego stan zostaje odczytany przez komputer i przeliczony na stan napełnienia. Czas bramkowania można uzyskać stosując albo zewnętrzny generator, albo za pomocą programowo realizowanej pętli opóźnienia. Jeśli oprogramowanie zawiera DOS-BASIC, to może być również w tym celu wykorzystany zegar wewnętrzny komputera. Obie dodatkowe linie sterujące są potrzebne do tego, aby sprawdzić moment zakończenia czasu bramkowania oraz do zerowania licznika przed rozpoczęciem następnego cyklu.

Ponieważ częstotliwość generatora w dużym stopniu zależy od wymiarów zbiornika oraz od własności cieczy, wspomniana metoda potraktowana jest tu jedynie jako zachęta do własnych eksperymentów.

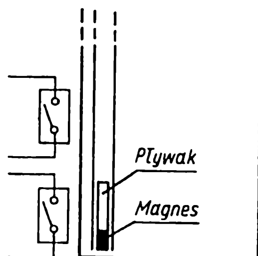
Teraz zostanie opisana realizacja innej metody.

Rysunek 2.20 przedstawia zasadę pomiaru. W rurze z tworzywa sztucznego, umieszczonej wewnątrz zbiornika, znajduje się pływak z magnesem. Naprzeciwko rury, na zewnątrz zbiornika, są umieszczone kontakty odczytowe. Są to zestyki[1], które znajdują się w rurkach szklanych wypełnionych gazem obojętnym, a włączane są przez pole magnetyczne. Jak długo pływający magnes znajduje się naprzeciwko zestyku, ten pozostaje zamknięty. Jest to więc prosty sposób kontroli stanu zapełnienia zbiornika, niezależny od rodzaju cieczy. Dodatkową zaletą jest tu brak galwanicznego połączenia między elementami zbiornika a układem pomiarowym.

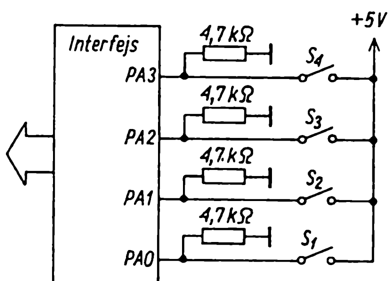
Na rysunku 2.21 pokazano układ pomiarowy, a w tablicy poniżej możliwe kombinacje sygnałów na liniach portu A.

Stopień wypełnienia	Odczytana kombinacja bitów	Odpowiadająca jej liczba dziesiętna
Minimum	0001	1
Norma 1	0010	2
Norma 2	0100	4
Maksimum	1000	8
Stan pośredni	0000	0

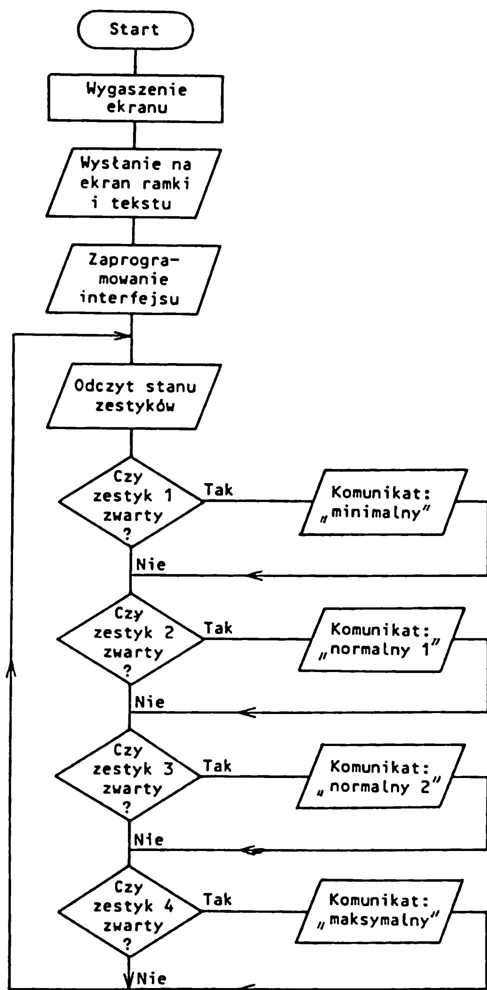
[1] Są to tzw. kontaktrony (przyp. tłum.).



Rys. 2.20. Zasada dyskretnego pomiaru stanu wypełnienia zbiornika



Rys. 2.21. Schemat układu do dyskretnego pomiaru stanu wypełnienia zbiornika



Rys. 2.22. Sieć działań programu "POMIAR WYPEŁNIENIA"

Program

Sieć działań programu "POMIAR WYPEŁNIENIA" pokazano na rys. 2.22. Po zaprogramowaniu interfejsu rozpoczyna się odczytywanie stanu zestyków poprzez port A. Jeżeli pływak znajduje się przypadkiem pomiędzy zestykami, tzn. żaden z nich nie jest zwarty, to odczytane zostaje zero. W takim przypadku na ekranie pozostaje poprzedni wynik pomiaru. W ten sposób nie może pojawić się żadne niedozwolone wskazanie.

Na rysunku 2.23 przedstawiono odpowiedni program napisany w języku Basic. Początkowa część służy do przygotowania ekranu monitora. W linii 210 zostaje zaprogramowany interfejs, a w linii 230 następuje odczyt stanu zestyków. Następnie odbywa się sprawdzenie i wysłanie na ekran monitora odpowiedniej informacji, jeśli któryś z zestyków był zwarty. W przeciwnym przypadku odczytane zero spowoduje, że poprzedni komunikat zostanie na ekranie. Stan ten będzie utrzymywał się tak długo, jak długo komputer nie odczyta wartości różnej od zera.

```
100 REM **** PROGRAM :"POMIAR WYPELNIENIA"        ****
110 REM **** NAZWA PROG.: "WYPELN"                ****
120 CLS
130 REM LINIE 140-190 : RAMKA NA EKRAN
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) SET (127,Y)
190 NEXT Y
200 PRINT@ 328;"P O M I A R   W Y P E L N I E N I A";
210 OUT 7,153 : REM ZAPROGRAMOWANIE INTERFEJSU
220 PRINT @ 328,"WYPELNIENIE "
230 H = INP(4) : REM ODCZYT STANU WYPELNIENIA
240 IF H = 1 PRINT @660,"MINIMALNE ";
250 IF H = 1 PRINT @660,"NORMALNE 1";
260 IF H = 1 PRINT @660,"NORMALNE 2";
270 IF H = 1 PRINT @660,"MAKSYMALNE";
280 GOTO 220
290 END
```

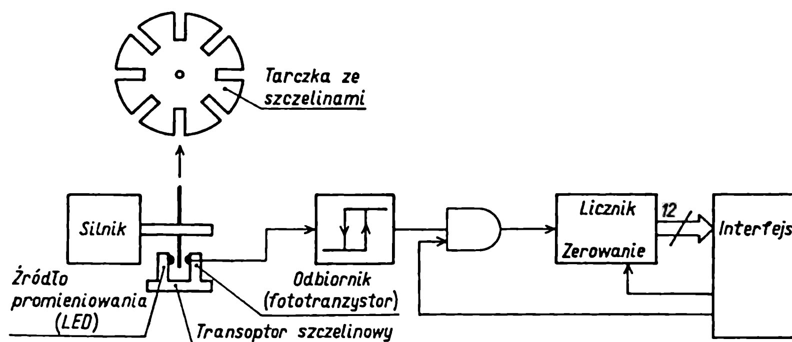
Rys. 2.23. Program "POMIAR WYPEŁNIENIA"

Program można łatwo rozszerzyć wstawiając następne linie z instrukcją IF dla sprawdzenia większej liczby zestyków (i poziomów cieczy). Pod względem sprzętowym jeden port umożliwia dołączenie ośmiu kontaktronów. Jeżeli wymagana jest większa liczba poziomów, to musi zostać użyty następny port[1)].

[1)] Konieczna jest wtedy również większa zmiana w programie (przyp. tłum.).

2.1.6. Pomiar prędkości obrotowej

Możliwe są dwie metody pomiaru prędkości obrotowej silnika. Pierwsza z nich polega na sprzężeniu prądnicy tachometrycznej z wałem silnika. Prądnica wytwarza napięcie proporcjonalne do prędkości obrotowej wału. Napięcie to po przetworzeniu na sygnał cyfrowy jest przesyłane do komputera. Druga metoda zakłada użycie tarczki ze szczelinami. Tarczka ta, o kształcie krążka, zostaje zamocowana na osi silnika. Na jej obrzeżu znajdują się szczeliny lub nacięcia. Brzeg tarczki jest wprowadzony w szczelinę transoptora szczelinowego, który składa się z diody emitującej promieniowanie widzialne lub podczerwone i fototranzystora jako odbiornika (rys. 2.24). Impulsy wytwarzane przez obracającą się z wałem silnika tarczkę są dostarczane w określonym czasie pomiaru — zwanym też czasem bramkowania — na wejście licznika. Czas bramkowania oraz niezbędne przed każdym cyklem zliczania impulsy zerowania licznika są dostarczane z komputera. Przed każdym wyzerowaniem licznika następuje odczyt jego stanu i wskazanie prędkości obrotowej.



Rys. 2.24. Pomiar prędkości obrotowej za pomocą tarczki ze szczelinami

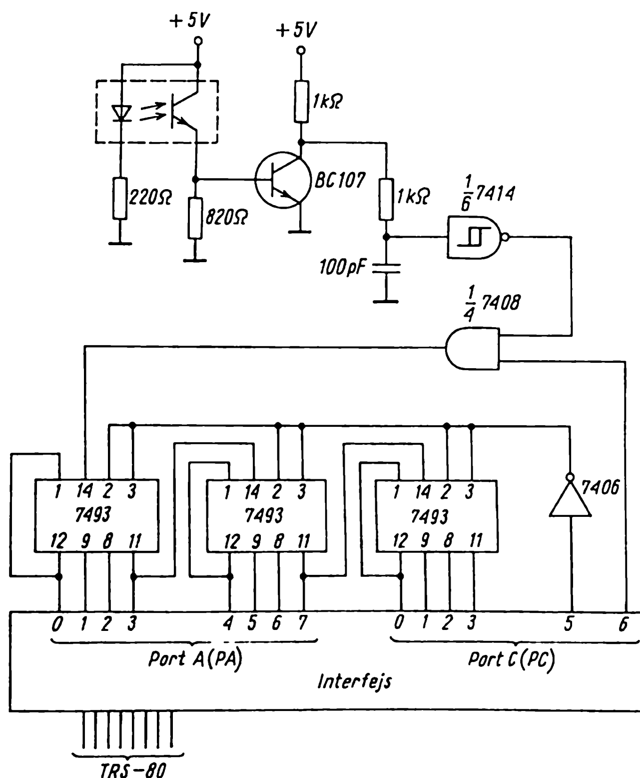
Ta metoda pomiaru daje się bardzo łatwo zrealizować, ponieważ sygnał wyjściowy otrzymuje się wprost w postaci cyfrowej. Z tego powodu znajduje ona zastosowanie w podanym niżej rozwiązaniu.

Parametry

- pomiar prędkości obrotowej w zakresie od 500 do 6000 obr/min,
- pomiar za pomocą tarczki z szesnastoma szczelinami,
- korekta wskazania w przypadku zatrzymania silnika,
- ostrzeżenie przy przekroczeniu maksymalnej prędkości obrotowej.

Sprzęt (układ pomiarowy)

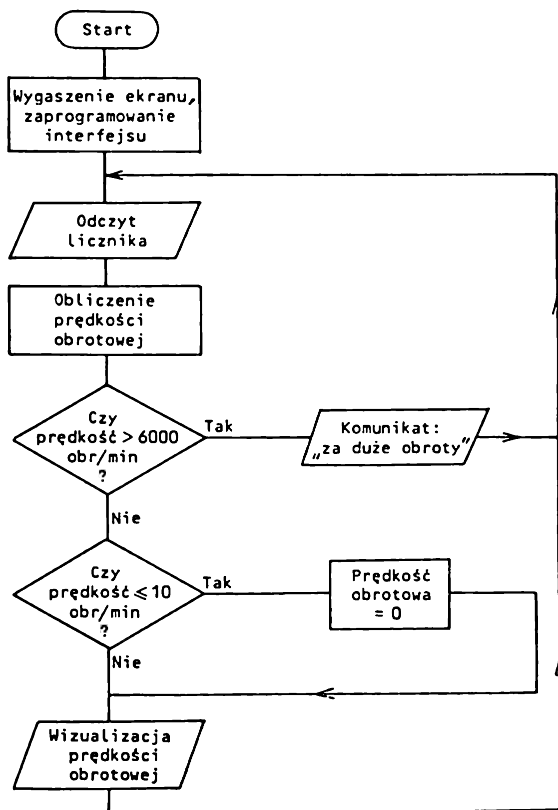
Układ pomiarowy pokazano na rys. 2.25. Wytwarzana przez diodę emitującą transoptora szczelinowego wiązka światła przechodząc przez szczelinę w tarczce oświetla fototranzystor. Ponieważ sygnał wyjściowy z fototranzystora, szczególnie przy dużych obrotach, wykazuje zbyt małą stromość zboczy, w tor włączono przerzutnik Schmitta. Filtr *RC* na wejściu tego przerzutnika służy do odfiltrowania impulsów zakłócających o szerokim widmie. Z wyjścia przerzutnika sygnał podawany jest poprzez bramkę NAND na wejście 12-bitowego licznika binarnego (trzy 4-bitowe układy SN 7493). Bramka NAND "przepuszcza" impulsy na wejście licznika w czasie pomiaru (czas bramkowania), tzn. w czasie, w którym komputer poprzez linię PC6 (6 — linia portu C) interfejsu wymusza na drugim wejściu bramki logiczną "1" (stan wysoki). Wszystkie układy licznika zerowane są równocześnie impulsem z wyjścia PC5 interfejsu, podawanym poprzez inwerter (SN 7406) na wejścia zerujące licznika (wyprowadzenie *2* i *3* układu 7493).



Rys. 2.25. **Układ do pomiaru prędkości obrotowej**

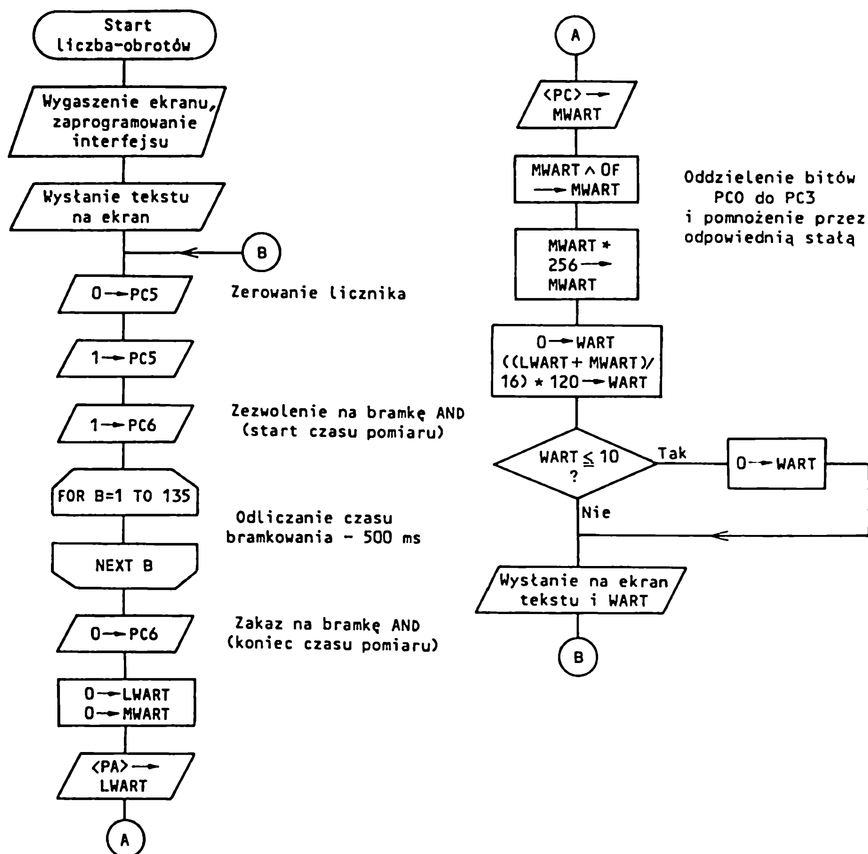
Program

Ogólna sieć działań programu "POMIAR PRĘDKOŚCI OBROTOWEJ" jest pokazana na rys. 2.26. Na początku zostaje zaprogramowany interfejs. Po odczytaniu i przeliczeniu wartości wskazywanej przez licznik, następuje wskazanie wartości chwilowej i ewentualne wyświetlenie ostrzeżenia, jeśli prędkość obrotowa jest zbyt duża. Jeśli rotor silnika się nie obraca, a szczelina tarczki przypadkowo trafia w strumień światła transoptora, to następuje błędny pomiar. Ten błąd, wynikający z zasady działania układu, jest korygowany programowo. Jeżeli zostanie stwierdzone, że prędkość obrotowa jest mniejsza niż 10 obr/min, to na ekranie monitora pokaże się wskazanie zerowe.



Rys. 2.26. Ogólna sieć działań programu "POMIAR PRĘDKOŚCI OBROTOWEJ"

Na rysunku 2.27 jest pokazana szczegółowa sieć działań programu. Można w niej wyróżnić symbole operacji wysłania sygnałów sterowania licznikiem oraz operacji odliczania czasu bramkowania (500 ms), za pomocą pętli opóźnienia. Ponieważ zmierzona liczba obrotów jest odczytywana w dwóch



Rys. 2.27. Szczegółowa sieć działań programu "POMIAR PRĘDKOŚCI OBROTOWEJ"

etapach, wartość najbardziej znaczących czterech bitów (bity od 8 do 11) odczytana poprzez port C musi być odpowiednio pomnożona przez 256 i dodana do odczytanej poprzez port A wartości liczby 8-bitowej.

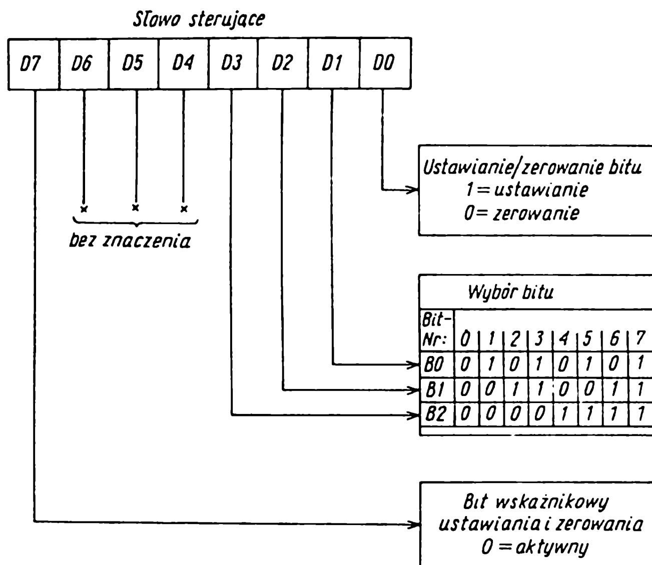
Na rysunku 2.28 pokazano program napisany w języku Basic. Po zaprogramowaniu interfejsu w linii 130 następuje przygotowanie graficzne ekranu za pomocą odpowiedniej podprocedury (linie 330-380). Potem następuje wyzerowanie licznika impulsów — linia 160 (wysłanie "H" i następnie "L" na PC5) i zezwolenie na zliczanie (początek czasu pomiaru) poprzez PC6 — linia 170. Po pętli opóźnienia (0,5 s) zliczanie zostaje przerwane (wysłanie "L" na PC6), a następnie zostaje odczytany i przetworzony stan licznika (linie 210—270). Ponieważ wykorzystuje się jedynie cztery mniej znaczące bity portu C, muszą być one poprzez operację maskowania oddzielone od pozostałych (iloczyn logiczny z

```
100 REM **** PROGRAM :"POMIAR PREDKOSCI OBROTOWEJ          ****
110 REM **** NAZWA PROG.:"OBRNAMIN"                        ****
120 CLS
130 OUT 7,145
140 GOSUB 340
150 PRINT@328,"P O M I A R   P R E D K O S C I   O B R O T O W E J ";
160 OUT 7,10 : OUT 7,11
170 OUT 7,13 :                     REM  POCZATEK ZLICZANIA (ZEZWOLENIE)
180 FOR B = 1 TO 135 : NEXT B :    REM  500 MILISEC. ZLICZANIA IMPULSOW
190 OUT 7,12 :                     REM  ZLICZANIE ZAKONCZONE (ZAKAZ)
200 LWART = 0 : MWART = 0:
210 LWART = INP(4)  :              REM  ODCZYT 8 MNIEJ ZNACZ. BITOW LICZNIKA
220 MWART = INP(6)  :              REM  ODCZYT 4 BRDZ. ZNACZ. BITOW LICZNIKA
230 MWART = MWART AND 15  :        REM  WYDZIELENIE 4 MNIEJ ZNACZ. BITOW PORTU
 C
240 MWART = MWART * 256  :         REM  KOREKCJA WARTOSCI TYCH 4 BITOW
250 WART = ((LWART+MWART)/16)*120  : REM  OBLICZENIE LICZBY OBROT. NA MIN
260 IF WART > 6000 PRINT@584, "UWAGA!  ZA DUZE OBROTY";
270 IF WART <=10 THEN WART = 0   :  REM  USTAWIENI ZEROWEGO WSKAZANIA
280                                 REM  JESLI TARCZKA NIERUCHOMA
290 PRINT @648,"LICZBA OBROTOW NA MIN. ";
300 M$="#####,####"
310 PRINT @720,USING M$;WART;
320 GOTO 160
330 REM  PODPROCEDURA : RAMKA NA EKRAN
340 FOR X = 0 TO 127
350 SET (X,1) : SET (X,47) : NEXT X
360 FOR Y = 1 TO 47
370 SET (0,Y) : SET (1,Y) : SET (126,Y) : SET(127,Y)
380 RETURN
390 REM   T A B E L A   Z M I E N N Y C H
400 REM   ==============================
410 REM   LWART      ODCZYTANA WART. 8 MNIEJ ZNACZ. BITOW LICZNIKA
420 REM   MWART      ODCZYTANA WART. 4 BARDZ.ZNACZ. BITOW LICZNIKA
430 REM   WART       AKTUALNA LICZBA OBROTOW NA MINUTE
```

Rys. 2.28. Program "POMIAR PRĘDKOŚCI OBROTOWEJ"

liczbą 1111_2[1] = 15_{10}[1] w linii 230). Z pojęciem maskowania można się bliżej zapoznać w p. 2.2.1.

W liniach 160, 170 i 190 wykorzystana została nie omawiana jeszcze właściwość układu 8255. Otóż każdy z ośmiu bitów portu C może być osobno ustawiony i zerowany przez wysłanie jednego słowa sterującego do rejestru sterowania. Na rysunku 2.29 jest pokazany format słowa sterującego ustawie-



Rys. 2.29. Format słowa sterującego ustawieniem i zerowaniem bitów portu C układu 8255

niem i zerowaniem bitów. Jeśli chcemy na przykład ustawić na jedynkę bit 5 w porcie C (PC5), to należy do rejestru sterowania wysłać 0000 1011_2 = 11_{10}. Użyty w naszym przypadku interfejs wymaga więc następującej instrukcji w Basicu:

OUT 7, 11

Jeżeli zostanie zastosowana tarczka o innej niż 16 liczbie szczelin, to należy 16 w linii 250 zastąpić aktualną liczbą szczelin. Czynnik 120 bierze się stąd, że prędkość obrotowa jest podawana w obrotach na minutę (60), a czas pomiaru wynosi 0,5 s (×2). Jeśli zostanie wybrany inny czas pomiaru (zmiana w pętli opóźnienia — linia 180), to trzeba tę liczbę odpowiednio skorygować.

[1] Indeks przy liczbie informuje, w jakim systemie liczbowym została ona zapisana (przyp. tłum.).

Ulepszenia

W przypadku tanich silników prądu stałego prędkość obrotowa podlega niestety znacznym wahaniom, zwłaszcza przy małych obrotach. Prowadzi to do migotania wskazania. Aby uzyskać stabilne wskazanie, zaleca się uśredniać wyniki dwóch lub trzech pomiarów i dopiero uśrednioną wartość pokazywać na ekranie.

2.2. Pomiary wielkości szybkozmiennych

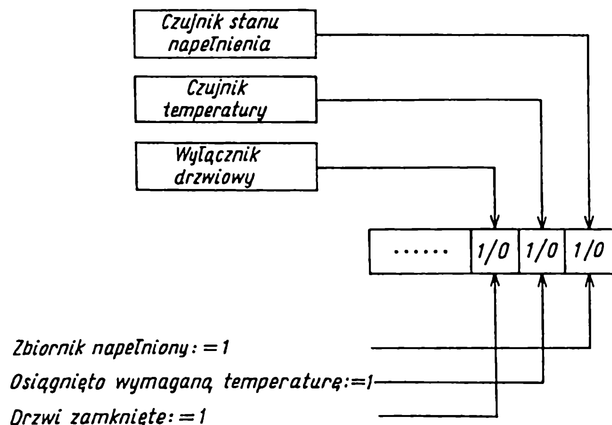
Komputery wyposażone w Basic, o ile wykorzystują interpreter, są stosunkowo wolne i dlatego nieprzydatne do pomiarów wielkości szybkozmiennych za pomocą programów napisanych w tym języku. Wprawdzie pojedyncza instrukcja wejścia (INP-) wymaga tylko kilku milisekund, ale nią samą niczego nie można zdziałać. Włącza się więc ją w program pobierania danych o długości czterech lub pięciu instrukcji, który odczytane dane przeznaczone do dalszego przetwarzania musi umieścić w pamięci. To wymaga już od 10 do 20 milisekund na jedną daną pomiarową. Jeżeli jednak korzysta się z dużego programu, w którym przetwarzanie danej następuje każdorazowo po jej pobraniu, to liczba próbek (odczytów wartości mierzonej) w ciągu sekundy spada do ok. 2—4. Dla przebiegów, których szybkość zmian liczona jest w setnych częściach sekundy, jest to o wiele za wolno. Niemożliwe jest także wykorzystanie maksymalnej szybkości działania przetwornika, a więc próbkowanie (i zobrazowanie kształtu) sygnałów szybkozmiennych ani obróbka przebiegów impulsowych o dużych częstotliwościach (patrz rozdz. 6).

Pomocą może być w takim przypadku kompilator Basicu, który tłumaczy program napisany w języku o orientacji problemowej na kody wewnętrzne urządzenia (kody maszynowe). Inna możliwość polega na tym, że programuje się w języku asemblera lub wprost w kodach wewnętrznych komputera (kody heksadecymalne). Jeśli mamy do dyspozycji program asemblera, to pracować jest trochę łatwiej, ponieważ instrukcje (języka) asemblera (tzw. mnemoniki) zapamiętuje się znacznie szybciej, a posługiwanie się nimi jest wygodniejsze niż rozkazami maszynowymi w kodach heksadecymalnych. Należy jednakże wziąć pod uwagę to, że operacje arytmetyczne jak mnożenie, dzielenie czy wyciąganie pierwiastka, nastręczają na tym poziomie programowania znaczne trudności. Trzeba więc przyjąć rozsądny kompromis. Te procedury programu, które wymagają krótkiego czasu wykonania, a więc te, których zadaniem jest zbieranie próbek wielkości mierzonej, pisze się w języku asemblera. Pozostałe części programu można już napisać w Basicu, a procedury w asemblerze wywoływać z programu napisanego w Basicu. W ten sposób można z jednej strony uzyskać znaczne zwiększenie szybkości działania (w dziedzinie kodów maszynowych: 300 do 1000 razy), a z drugiej — zmniejszyć wysiłek poświęcony na napisanie programu.

Ponieważ ta książka poświęcona jest przede wszystkim zastosowaniom Basicu w technice pomiarowej, sterowania i regulacji, nie będziemy zagłębiać się w technikę programowania w asemblerze. Przykłady programów w języku asemblera zostaną tu jednak przytoczone, głównie dla tych czytelników, którzy bądź opanowali ten język, bądź też gotowi są poznać tajniki programowania w języku wewnętrznym ze względu na jego ogromną zaletę, jaką jest szybkość. Trzeba jednak podkreślić, że procedury pisane w asemblerze powinny być tak krótkie, jak to jest tylko możliwe i powinny być wywoływane z programu napisanego w Basicu. Przetwarzanie danych pomiarowych następuje ostatecznie w programie napisanym w Basicu.[1]

2.2.1. Podstawy techniki programowania

Przy programowaniu operacji związanych z techniką pomiarów, sterowania i regulacji, duże znaczenie ma testowanie sygnałów statusu albo sygnałów współpracy z jednej strony, a wysyłanie sygnałów sterujących z drugiej. Na ogół informacje są odczytywane, przetwarzane i wysyłane przez komputery jedynie w postaci słów, czyli bloków 8- lub 16-bitowych. Dla każdego sygnału statusu, albo sygnału sterującego, musiałoby więc być użyte jedno słowo (w przypadku TRS-80: jeden bajt), czyli jeden cały port. Byłaby to więc duża rozrzutność. W "eleganckich" systemach pomiarowych nie wystarczyłyby nawet trzy stojące do dyspozycji porty. Wprawdzie można by przewidzieć dołączenie dalszych ukła-



Rys. 2.30. Zestawienie kilku różnych informacji o stanie pralki automatycznej w postaci jednego słowa statusu

[1] W dalszej części rozdziału mowa będzie o programach napisanych w języku Basic oraz w języku asemblera. Aby nie używać zbyt długich opisowych nazw, używane będą określenia skrócone, a więc odpowiednio program (Basic) i program (asembler).

dów wejścia/wyjścia, prościej jest jednak "upakować" więcej informacji w jednym słowie statusu lub w słowie sterującym. Na rysunku 2.30 zilustrowano tę metodę na przykładzie pralki automatycznej. Wymagane selektywne testowanie poszczególnych bitów słowa statusu jest realizowane po jego odczytaniu poprzez instrukcję Basicu bądź asemblera. Zwłaszcza język asemblera odpowiada tym potrzebom, ze względu na dużą liczbę dogodnych instrukcji, podczas gdy testowanie za pomocą instrukcji Basicu jest w dodatku bardzo czasochłonne.

Jeśli więc istotna jest szybkość działania, to powinno się poniższe procedury napisać w języku asemblera.

Maskowanie bitów

Jeśli w odczytanym słowie statusu ma być selektywnie sprawdzony jeden z bitów, wszystkie pozostałe trzeba uczynić "niewidocznymi". Taka operacja jest nazywana *maskowaniem*. Tak jak maska pozwala dostrzec z oryginału (twarzy) jedynie niezbędne minimum, powiedzmy wymarzony wizerunek, tak tutaj z oryginału (słowa statusu) powinien być widoczny jedynie pożądany fragment, a wszystkim pozostałym bitom winna zostać przypisana określona wartość — najczęściej zero.

Tak stanie się, jeśli wykonamy operację AND na słowie statusu i pewnej stałej (masce): to znaczy $X \wedge 1 = X$ oraz $X \wedge 0 = 0$, gdzie X oznacza zero lub jedynkę.

	1/0	.....	1/0	1/0	1/0	słowo statusu
∧	0	.....	0	0	1	maska
	0	.....	0 .	0	1/0	wynik

Ponieważ w tym przypadku został "przepuszczony" z całego słowa tylko jeden bit, sprawdzenie jest bardzo proste: jeśli wynik operacji AND (maskowania) ma wartość różną od zera, bit był ustawiony, w przeciwnym razie bit był wyzerowany. Tak więc, jeśli trzeba sprawdzić bit 1, jako maski użyjemy: 0 0 — — 0 1 0.

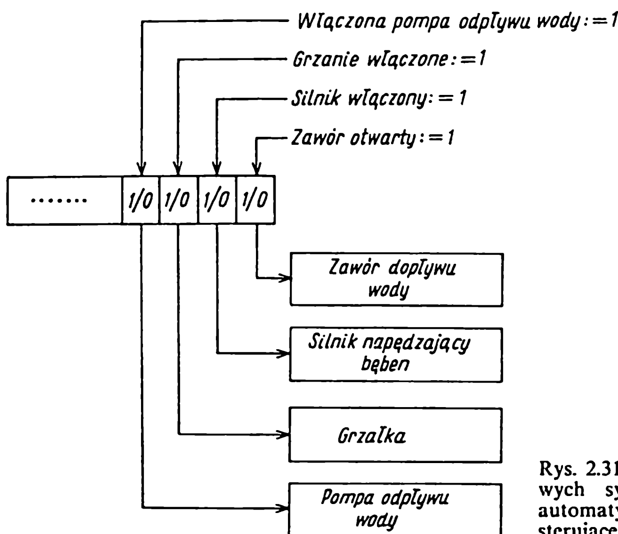
W ten sam sposób można sprawdzać grupy bitów. Przypuśćmy, że w pralce automatycznej, dla której słowo statusu pokazano na rys. 2.30, sprawdzony ma być przełącznik programów prania. Jeśli mamy wybrać jeden z czterech programów, to trzeba sprawdzić dwa dodatkowe bity, a przebieg sprawdzania będzie następujący:

	1/0	.....	1/0	1/0	1/0	1/0	1/0	słowo statusu
∧	0	.....	1	1	0	0	0	maska
	0	.....	1/0	1/0	0	0	0	wynik

Po podzieleniu wyniku przez 8 bądź przesunięciu słowa o trzy miejsca w prawo, otrzymamy właściwy numer programu.

Manipulowanie bitami

Maskowanie było istotne w przypadku sprawdzania odczytanej informacji o stanie urządzenia. Odwrotnie — jeśli chcemy wysyłać do urządzenia sprzężonego z komputerem sygnały sterowania, to trzeba je "poskładać" w jedno słowo sterujące. Na rysunku 2.31 pokazano przykład sterowania pralką automatyczną. W tym przypadku musi istnieć możliwość ustawienia lub zerowania każdego z bitów (każdej linii sterującej), niezależnie od pozostałych. Takie operacje mogą być przeprowadzane również dzięki instrukcjom logicznym.



Rys. 2.31. Zestawienie różnych możliwych sygnałów sterujących pralkę automatyczną w postaci jednego słowa sterującego

Aby jednemu z bitów nadać wartość 1, wykonuje się operację OR na słowie sterującym i pewnej stałej:

	1/0	.....	1/0	1/0	poprzednie słowo sterujące
v	0	.....	0	1	stała
	1/0	.....	1/0	1	wynik — nowe słowo sterujące

Ponieważ 0 v X = X oraz 1 v X = 1, więc bit najmniej znaczący (bit 0) niezależnie od jego poprzedniej wartości został ustawiony na "1" logiczną. Pozostałe bity (sygnały sterujące) zatrzymają swoje poprzednie wartości. Jeśli bit 1 ma mieć wartość logiczną "1", to należy użyć stałej 0——— 0 1 0.

Selektywne zerowanie poszczególnych bitów słowa sterującego realizuje się za pomocą instrukcji AND i odpowiedniej stałej:

	1/0	.....	1/0	1/0	poprzednie słowo sterujące
∧	1	.....	1	0	stała
	1/0	.....	1/0	0	wynik — nowe słowo sterujące

Ponieważ 1 ∧ X = X oraz 0 ∧ X = 0, więc najmniej znaczący bit niezależnie od swojej wartości został wyzerowany. Wszystkie pozostałe bity zachowują swą poprzednią wartość. Jeśli ma być wyzerowany bit 1, to należy użyć stałej 1 — — — 1 0 1.

Ustawianie i zerowanie pojedynczych bitów powtarza się zwykle w procesie sterowania bardzo często. W związku z tym firma Intel przewidziała w równoległym układzie wejścia/wyjścia 8255 możliwość niezależnego wpływu na poszczególne bity portu C. Bliższych informacji na ten temat można zaczerpnąć z p. 2.1.6.

2.2.2. Oscyloskop przebiegów małej częstotliwości

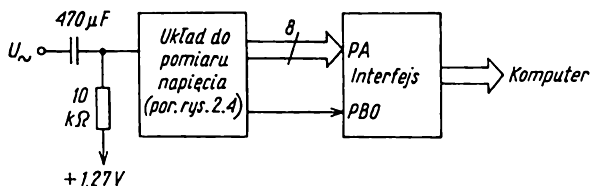
Jako przykład szybkiego próbkowania sygnału okresowego za pomocą przetwornika A/C, zostanie omówiona procedura próbkowania, zapamiętania i przedstawienia w postaci graficznej przebiegów oscylacyjnych o niezbyt dużych częstotliwościach. Sterowanie systemem pomiarowym oraz odczyt zmierzonych wartości chwilowych są realizowane za pomocą programu napisanego w asemblerze. Program ten wpisuje 128 próbek mierzonego sygnału do pamięci komputera, tworząc w ten sposób tablicę wartości chwilowych. Do tej tablicy ma następnie dostęp program (Basic), za pomocą którego przetwarza się otrzymane dane i wysyła wyniki na ekran.

Parametry

- automatyczne pobieranie 128 próbek w odstępie ok. 1,2 ms,
- maksymalna amplituda napięcia przemiennego: 1,27 V (można ją dowolnie powiększać przez dobór rezystancji szeregowej),
- wartość skuteczna napięcia wejściowego: max 0,9 V,
- maksymalna częstotliwość ok. 80 Hz,
- graficzne przedstawienie przebiegu na ekranie monitora.

Sprzęt (układ pomiarowy)

Na rysunku 2.32 pokazano schemat blokowy układu pomiarowego. Ponieważ chodzi tu w zasadzie o omówiony w p. 2.1.1 układ do pomiaru napięcia, nie ma potrzeby podawania jego szczegółowego schematu ideowego.



Rys. 2.32. Schemat blokowy układu wejściowego cyfrowego oscyloskopu

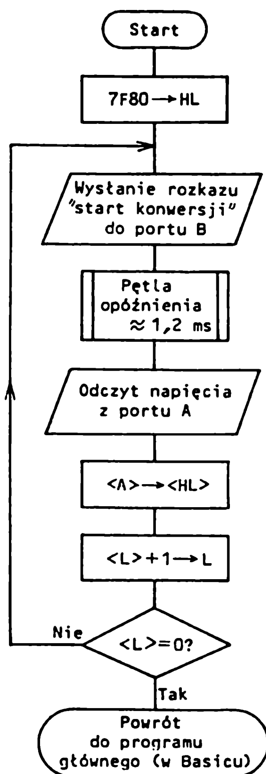
Ze względu na to, że użyty przetwornik A/C może przetwarzać jedynie napięcia dodatnie, trzeba w przypadku pomiaru napięcia przemiennego zastosować taką wstępną polaryzację, aby na wejściu przetwornika nie pojawiały się napięcia ujemne. Jak pokazano na rys. 2.32, napięcie polaryzacji wstępnej wynosi 1,27 V.

Program

Ogólna sieć działań podprogramu napisanego w asemblerze jest pokazana na rys. 2.33. Po załadowaniu do rejestrów HL początkowego adresu obszaru pamięci, do którego zostaną wprowadzone dane pomiarowe (tutaj $7F80_{16}$)[1)], zostaje wysłany rozkaz startu konwersji (poziom "L" a po nim "H" na linię 0 w porcie B, lub w programie 00, a następnie 01 do portu B). Potem następuje pętla opóźnienia ok. 1,2 ms (to jest nieco więcej niż maksymalny czas konwersji, który wynosi 1 ms). Czas opóźnienia jest określony wielkością stałej wprowadzonej do rejestru C. Po czasie opóźnienia, odczytana poprzez port A cyfrowa wartość chwilowa napięcia wejściowego zostaje zapisana w komórce pamięci RAM o adresie przechowywanym w parze rejestrów HL — tutaj na początku 7F80 — (adresowanie pośrednie przez rejestr). Teraz adres zostaje zwiększony o jeden i program sprawdza, czy zostało już załadowanych 128 komórek pamięci. Jeśli tak, to następuje powrót do programu głównego (Basic), a jeśli nie — dalsze pobieranie próbek. Ograniczenie liczby próbek do 128 wynika stąd, że TRS-80 ma bardzo skromną pseudografikę, która umożliwia przedstawienie jedynie 128 punktów w kierunku poziomym. W przypadku komputera o większej rozdzielczości liczba musi być odpowiednio zmieniona. Adres początkowy obszaru pamięci RAM przeznaczonego na tablicę wartości przebiegu — $7F80_{16} = 32640_{10}$ — wynika z zastosowania pewnego triku, dzięki któremu skraca się program (asemblera) i zwiększa szybkość przetwarzania. Po 128 przebiegach "koniec tablicy" dociera do komórki o adresie 7FFF. Rejestr L zawiera więc liczbę FF i następna inkrementacja "zwiększy" jego zawartość na 00. Dzięki temu możliwe jest proste sprawdzenie "czy zero" za pomocą skoku warunkowego (JNZ — skok jeśli nie zero).

---

1) Indeks informuje, że liczba jest zapisana w systemie szesnastkowym (heksadecymalnym) (przyp. tłum.).



Rys. 2.33. Ogólna sieć działań programu "OSCYLOSKOP CYFROWY" (w języku asemblera)

Adresy (Hex)	Mnemoniki	Kody maszynowe (Hex)	Kody maszynowe (Dec)
7F00	LXI HL,7F80	21	33
01		80	128
02		7F	127
03	MVI A,00	3E	62
04		00	0
05	OUT 05	D3	211
06		05	5
07	MVI A,01	3E	62
08		01	1
09	OUT 05	D3	211
0A		05	5
0B	MVI C,96	0E	14
0C		96	150
0D	DCR C	0D	13
0E	JNZ 7F0D	C2	194
0F		0D	13
10		7F	127
11	IN 04	DB	219
12		04	4
13	MOV M,A	77	119
14	INR L	2C	44
15	JNZ 7F03	C2	194
16		03	3
17		7F	127
18	RET	C9	201

Rys. 2.34. Program "OSCYLOSKOP CYFROWY" napisany w asemblerze mikroprocesora 8080

Na rysunku 2.34 jest pokazany program w języku asemblera łącznie z kodami języka wewnętrznego w postaci heksadecymalnej i dziesiętnej. Do czego potrzebne są kody w postaci dziesiętnej wyjaśnione zostanie później. W programie tym zostały użyte instrukcje asemblera (mnemoniki) mikroprocesora Intel 8080, ponieważ jest to najbardziej znany mikroprocesor, a do tego częściowo kompatybilny z mikroprocesorem Z80 zastosowanym w TRS-80 (zbiór instrukcji mikroprocesora Intel 8080 jest podzbiorem instrukcji mikroprocesora Z80). Wprawdzie dla obydwu procesorów instrukcje o jednakowym działaniu mają różne mnemoniki, ale zgadzają się dokładnie kody języka wewnętrznego (kody maszynowe) potrzebne do stworzenia kombinowanego programu w Basicu i asemblerze.

Dla specjalistów od Z80 został przedstawiony na rys. 2.35 program w asemblerze mikroprocesora Z80. Przy pisaniu tych programów założono dla uproszczenia, że amplituda mierzonego napięcia nie będzie wykraczać poza zakres pracy przetwornika i dlatego nie sprawdza się sygnału statusu.

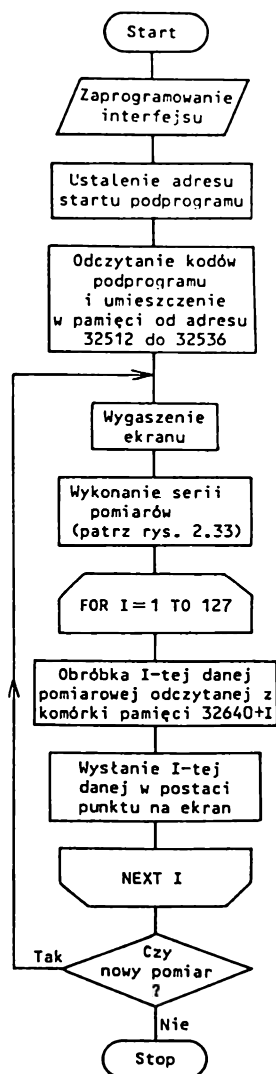
Adresy początkowe obydwu programów (asemblera) — 32512_{10} lub tablicy wartości sygnału — 32640_{10}, można w zasadzie wybrać dowolnie. Zależą one jedynie od wielkości pamięci RAM, trzeba się więc na początku upewnić, czy obszary pamięci o takich adresach są dostępne w waszym komputerze. Powinny one też być tak wybrane, aby te części programu znalazły się w obszarze pamięci o wysokich adresach. Można je wówczas łatwo zabezpieczyć przez skasowanie, np. wskutek zapisania w tym miejscu innej części programu w Basicu. Wystarczy w tym celu bezpośrednio po włączeniu komputera, kiedy na ekranie pojawi się

Adresy (Hex)	Mnemoniki	Kody maszynowe(Hex)	Kody maszynowe (Dec)
7F00	LD HL,7F80	21	33
01		80	128
02		7F	127
03	LD A,0	3E	62
04		00	0
05	OUT 05,A	D3	211
06		05	5
07	LD A,1	3E	62
08		01	1
09	OUT 05,A	D3	211
0A		05	5
0B	LD C,96	0E	14
0C		96	150
0D	DEC C	0D	13
0E	JP NZ,7F0D	C2	194
0F		0D	13
10		7F	127
11	IN A,04	DB	219
12		04	4
13	LD (HL),A	77	119
14	INC L	2C	44
15	JP NZ,7F03	C2	194
16		03	3
17		7F	127
18	RET	C9	201

Rys. 2.35. Program "OSCYLOSKOP CYFROWY" napisany w asemblerze mikroprocesora Z80

pytanie "MEMORY SIZE?", podać w postaci dziesiętnej adres, od którego będzie działało ograniczenie zapisu i poprzez wciśnięcie ENTER zaakceptować go. Przy powyżej wybranych adresach będzie to: 32512.

Program maszynowy odczytuje wyniki pomiarów z przetwornika 128 razy w odstępach ok. 1 ms i umieszcza go w buforze o pojemności 128 bajtów, którym jest fragment pamięci RAM o adresach od 7F80 do 7FFF. Po



Rys. 2.36. Sieć działań programu "OSCYLOSKOP CYFROWY"

zakończeniu tej procedury do bufora "sięgać" będzie program główny (napisany w Basicu).

Na rysunku 2.36 jest przedstawiona sieć działań programu głównego, a na rys. 2.37 — program napisany w Basicu. Podzielić go można w zasadzie na dwie części. Pierwsza z nich — do linii 180 — wprowadza do pamięci procedurę napisaną w asemblerze. Na początku zostaje określony adres startu tego programu. Znak "&H" przed adresem 7F00 sygnalizuje, że chodzi o liczbę w systemie szesnastkowym (heksadecymalnym). Zamiast tego można by także posłużyć się wprost systemem dziesiętnym, pisząc: DEFUSR 0(X) = 32512. Linia 180 (DATA — dane do wprowadzenia) zawiera ciąg instrukcji programu maszynowego, przy czym wszystkie kody instrukcji muszą być podane dziesiętnie (!). Z tego też powodu kody instrukcji w programie maszynowym podane zostały zarówno w kodach heksadecymalnych, jak też w zapisie systemu dziesiętnego.

W drugiej części programu głównego następuje wywołanie procedury asemblerowej (linia 200), a potem odczyt zapisanych w tablicy kolejnych wartości pomiarowych (linia 220). Po odjęciu przesunięcia i korekcji amplitudy (linia 230) następuje przeliczenie wartości chwilowych na współrzędne ekranowe (linia 240). Wybrano tu liczbę 23, ponieważ na ekranie TRS-80 mieści się 48 linii i linia zera przebiegu powinna leżeć mniej więcej pośrodku ekranu. W przypadku komputera o większej liczbie linii, wartość ta powinna być odpowiednio skorygowana. W końcu (linia 240) zostaje zapalony odpowiedni punkt na ekranie.

Należy zwrócić uwagę na to, że program główny (Basic) został napisany dla komputera ze stacją dysków przy użyciu języka DOS-BASIC. Jeżeli wykorzystuje się TRS-80 z językiem BASIC-Level II, to linie 130 i 200 muszą zostać zmienione. W tej wersji Basicu nie ma instrukcji DEFUSR. Zamiast tego adresy startu są wprowadzane do pamięci za pomocą instrukcji POKE pod adres 16526 (bardziej znaczący bajt adresu) oraz pod adres 16527 (bajt mniej znaczący). Ponieważ jedynie tu możliwe jest zapamiętanie adresu programu, wywołanie podprogramu maszynowego w linii 200 także brzmi inaczej: S = USR(0).

Z powodu wspomnianej już niskorozdzielczej grafiki komputera, przebieg oglądany na ekranie wygląda niezbyt ładnie. W przypadku komputera z grafiką o wysokiej rozdzielności (pełna grafika), można uzyskać przebiegi jak na ekranie oscyloskopu. Można też potraktować otrzymane punkty, jako co drugi, trzeci lub czwarty punkt krzywej i łączyć je programowo (BASIC) za każdym razem za pomocą odcinków prostoliniowych. W ten sposób można kilkakrotnie podwyższyć częstotliwość graniczną przedstawianej krzywej.

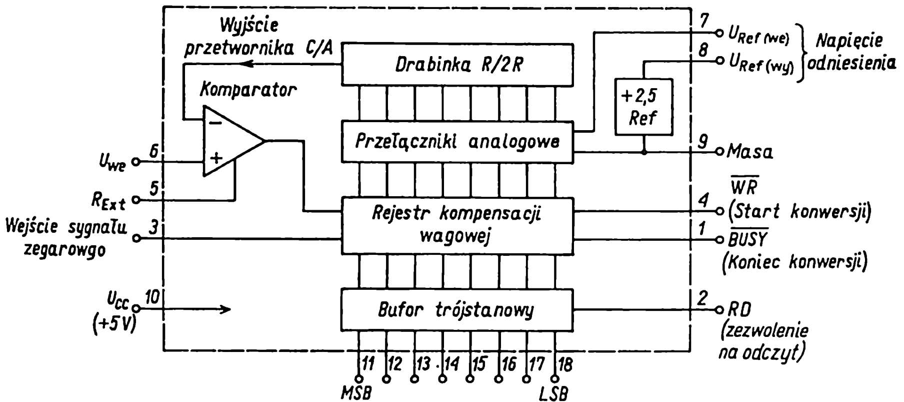
Pokazany na rys. 2.37 program nie zawiera żadnej funkcji, która odpowiadałaby znanej w oscyloskopie funkcji wyzwalania. Jeśli więc przewidujemy przeprowadzenie następnego cyklu pomiarowego, to trzeba się liczyć z tym, że nowa krzywa będzie się różnić od obserwowanej poprzednio, ponieważ faza przebiegu będzie przypadkowa.

```
100 REM **** PROGRAM :"OSCYLOSKOP CYFROWY"          ****
110 REM **** NAZWA PROG.:"OSCYLOS"                  ****
120 OUT 7,153 :                  REM ZAPROGRAMOWANIE INTERFEJSU
130 DEFUSR 0 = &H7F00     :      REM ADR. STARTU PROG. MASZYNOWEGO
140 FOR X = 32512 TO 32536 :     REM WCZYTANIE PROGRAMU MASZYNOWEGO
150 READ D :                     REM DO WYZNACZONEGO OBSZ. PAMIECI
160 POKE X,D
170 NEXT X
180 DATA 33,128,127,62,0,211,5,62,1,211,5,14,150,13,194,13,127,219,4,119,44,194,
3,127,201
190 CLS
200 S= USR0(X)
210 FOR I = 0 TO 127
220 A = PEEK(32640 + I)
230 AM = (A - 127)/5.5
240 Y = 23 - INT(AM)
250 SET(1,Y)
260 NEXT I
270 PRINT@ 30,"NOWY POMIAR ?  T/N";
280 INPUT A$
290 IF A$ = "T" GOTO 190
300 END
```

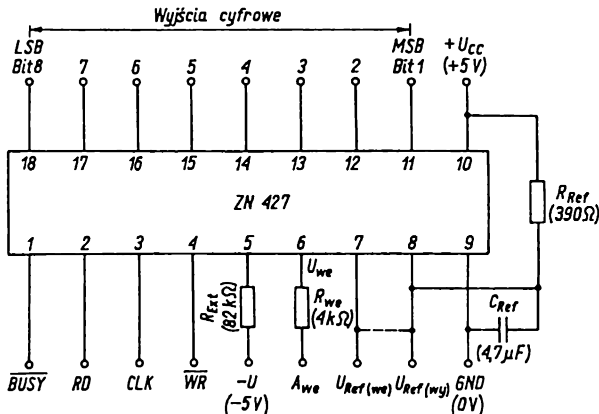
Rys. 2.37. Program "OSCYLOSKOP CYFROWY" napisany w języku Basic

Ulepszenia

Ponieważ pomiar wartości chwilowej jest realizowany za pomocą szybkiego programu maszynowego, oddzielnie od prezentacji graficznej, powolność programu głównego (Basic) staje się niezauważalna. Graniczna częstotliwość prezentowanych przebiegów nie zależy więc od szybkości części programu napisanej w Basicu, a tylko od podprogramu napisanego w asemblerze oraz od parametrów przetwornika A/C. Jeżeli opuścić w tym programie pętlę opóźnienia (adresy 7F0B—7F10) możliwe jest zmniejszenie odstępu pomiędzy pomiarami (pobraniem kolejnych próbek) do 40 µs. Jako przetwornik A/C — o ile wystarczająca jest rozdzielczość 8-bitowa — może być użyty układ ZN 427E firmy Ferranti, o czasie konwersji 15 µs (przy częstotliwości taktowania 600 kHz). Wymagane dla ZN 425 zewnętrzne elementy dodatkowe, jak wzmacniacz operacyjny i bramki, są w przypadku ZN 427E wbudowane wewnątrz układu.



Rys. 2.38. Schemat blokowy przetwornika ZN 427E



Rys. 2.39. Zewnętrzne połączenie ZN 427E

Schemat blokowy przetwornika przedstawiono na rys. 2.38, a podstawowy układ połączeń — na rys. 2.39.

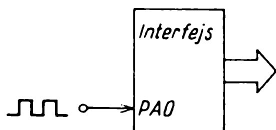
2.2.3. Pomiar czasu trwania impulsu

Parametry

- pomiar szerokości impulsu przy dowolnym współczynniku wypełnienia,
- maksymalna, dająca się zmierzyć, szerokość impulsu: ok. 1 s,
- dokładność pomiaru: 0,2% przy 100 ms, 2% przy 1 ms.

Sprzęt (układ pomiarowy)

Wymagany do tego pomiaru sprzęt daje się opisać w kilku słowach: składa się on wyłącznie z interfejsu zalecanego dla TRS-80 (rys. 2.40).



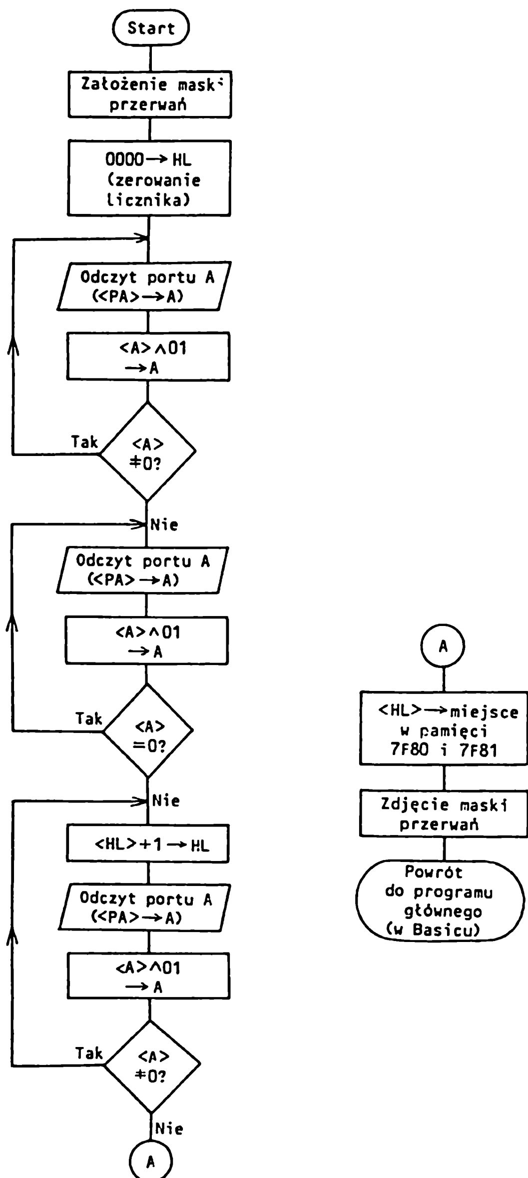
Rys. 2.40. Przyłączenie interfejsu w układzie do pomiaru szerokości impulsu

Program

Program do pomiaru szerokości impulsu składa się z dwu części: pierwszej — szybkiej podprocedury w asemblerze, za pomocą której jest przeprowadzany właściwy pomiar, a jego wynik pozostaje w pamięci, i drugiej — napisanej w Basicu, do której należy przetworzenie wyniku i wysłanie go na ekran.

Na rysunku 2.41 przedstawiono sieć działań procedury napisanej w asemblerze. Zasadniczą część tego programu stanowią trzy pętle. Na początku zostaje zablokowana obsługa przerwań (takie przerwania mogłyby przyjść np. od zegara programowego, który jest w systemie operacyjnym DOS, ewentualnie z klawiatury). Potem zostaje wyzerowana para rejestrów HL, która będzie tu wykorzystywana jako licznik. Teraz następuje pierwsza pętla, w której jest sprawdzane, czy na PA0 pojawił się stan wysoki (impuls). To selektywne testowanie odbywa się za pomocą operacji AND na akumulatorze (rejestr A) i stałej 01 (por. p. 2.2.1). Komputer pozostaje w tej pętli tak długo, jak długo na wejściu PA0 jest logiczna "1". Jest to niezbędne, ponieważ procedura zostaje wywołana przez program główny (w Basicu) w przypadkowym momencie, co może prowadzić do błędnych pomiarów. Jeśli mianowicie, program maszynowy zostanie wywołany w trakcie impulsu, to zmierzony zostanie czas trwania pozostałej jego części.

Druga pętla rozpocznie się jedynie wówczas, gdy pojawi się przerwa między impulsami. Komputer pozostanie w niej dopóty, dopóki nie rozpocznie się następny impuls. Przechodzi on wtedy do trzeciej pętli, w której dokonuje się



Rys. 2.41. Sieć działań programu "POMIAR SZEROKOŚCI IMPULSU" (w asemblerze)

właściwego pomiaru szerokości impulsu. Tak długo, jak długo trwać będzie impuls ("1" na PA0), komputer będzie w trzeciej pętli, a liczba obiegów pętli zostanie zliczona za pomocą pary rejestrów HL (16 bitów). Jeden obieg trwa ok. 20 μs, a ponieważ 16 bitów pozwala zliczać do 65535, maksymalna szerokość impulsu wynosi ok. 1,3 s. Natomiast przy bardzo krótkich impulsach rośnie błąd pomiarowy. Jeśli np. szerokość impulsu wynosi 50 μs (co przy stosunku czasu trwania impulsu do czasu przerwy równym 1:1 odpowiada częstotliwości mierzonej 10 kHz), to mogą nastąpić dwa albo trzy obiegi pętli. Spowoduje to zmiany wskazania o 30%, a więc średni błąd pomiarowy wynosi 15%.

Po opuszczeniu trzeciej pętli program wpisuje liczbę obiegów do pamięci RAM (komórki o adresach 7F80 i 7F81), zdejmuje maskę przerwań i skacze do programu głównego (Basic). Na rysunku 2.42 jest pokazany program napisany w asemblerze, w mnemonikach dla 8080 (maszynowe kody instrukcji dla Z80 są dokładnie takie same!).

Adres (hex.)	Mnemonika	Kod masz. (hex.)	Kod masz. (dzies.)	Objasnienia
7F00	DI	F3	243	Zal. maski przerwan
01	LXI HL,0000	21	33	Zerowanie licznika
02		00	0	
03		00	0	
04	IN 04	DB	219	Odczyt portu A
05		04	4	
06	ANI 01	E6	230	Maskowanie bitu 0
07		01	1	
08	JNZ 7F04	C2	194	Bit 0 = 1 ?
09		04	4	
0A		7F	127	
0B	IN 04	DB	219	Odczyt portu A
0C		04	4	
0D	ANI 01	E6	230	Maskowanie bitu 0
0E		01	1	
0F	JZ 7F0B	CA	202	Bit 0 = 0 ?
10		0B	11	
11		7F	127	
12	INX HL	23	35	Zwieksz licznik o 1
13	IN 04	DB	219	Odczyt portu A
14		04	4	
15	ANI 01	E6	230	Maskowanie bitu 0
16		01	1	
17	JNZ 7F12	C2	194	Bit 0 = 1 ?
18		12	18	
19		7F	127	
1A	SHLD 7F80	22	34	Przeslanie HL (stan licznika) do pam. o adr. 7F80 i 7F81
1B		80	128	
1C		7F	127	
1D	EI	FB	251	Zdjecie maski przer
1E	RET	C9	201	Powrot do programu glownego (BASIC)

Rys. 2.42. Program "POMIAR SZEROKOŚCI IMPULSU" napisany w asemblerze

```
100 REM **** PROGRAM :"POMIAR CZASU TRWANIA IMPULSU"****
110 REM **** NAZWA PROG.:"SZERIMP"                  ****
120 CLS
130 REM LINIE 140-190 : RAMKA NA EKRAN
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) SET (127,Y)
190 NEXT Y
200 PRINT@ 328;"P O M I A R   SZER.   I M P U L S U"
210 OUT 7,153 : REM ZAPROGRAMOWANIE INTERFEJSU
220 DEFUSR 0 = &H7F00
230 FOR X = 32512 TO 32542
240 READ D
250 POKE X,D
260 NEXT X
270 DATA 243,33,0,0,219,4,230,1,194,4,127,219,4,230,1,202,11,127,35,219,4,230,1,
194,18,127,34,128,127,251,201
280 S = USR0(X)
290 A = PEEK(32640)
300 B = PEEK(32641)
310 C = 256 * B + A
320 P = C/52.19
330 PRINT @648,"SZERROKOSC IMP. WYNOSI :";
340 M$ ="####.##"
350 PRINT @713, USING M$; P ;
360 PRINT" MILISEKUND";
370 FOR I = 1 TO 150 : NEXT I
380 GOTO 280
390 END
```

Rys. 2.43. Program "POMIAR SZEROKOŚCI IMPULSU" napisany w Basicu

Jeśli chodzi o adresy obszaru pamięci, pod którymi został umieszczony program oraz wartość mierzona, to mają one dokładnie takie samo znaczenie jak w przypadku wspomnianym w p. 2.2.2. Można je wybrać dowolnie i dlatego powinniście się Państwo upewnić, czy w waszym komputerze ten obszar pamięci jest do dyspozycji.

Na rysunku 2.43 jest przedstawiona część główna omawianego programu (napisana w Basicu). W pierwszym fragmencie następuje przygotowanie ekranu (linie 130-190) oraz wprowadzenie za pomocą polecenia DATA kodów podprocedury asemblerowej (linia 270). Podprocedura ta będzie wywoływana w linii 280. Zapamiętana w komórkach pamięci 32640 i 32641, zmierzona wartość zostanie następnie odczytana, a oba bajty dodane do siebie z uwzględnieniem ich wag (linie 290—310). Dzielenie, przeprowadzane w linii 320, przez stałą 52,19 określa dokładność pomiaru. Stała ta jest wyznaczana przez czas jednego przebiegu trzeciej pętli procedury pomiarowej (asembler). Ten ostatni powinien być indywidualnie sprawdzony dla każdego komputera. Do tego celu powinno się użyć wzorcowego przyrządu pomiarowego (np. oscyloskopu) i zmieniać stałą w linii 320 tak długo, jak długo wskazywana przez komputer szerokość impulsu nie zrówna się ze zmierzoną przyrządem wzorcowym.

Ulepszenia

W przypadku krótkich impulsów zakres pomiarowy jest ograniczony szybkością działania mikroprocesora. Tu niestety nie można uzyskać żadnej poprawy. Dla długich czasów ograniczenie wynika z tego powodu, że w rejestrach HL można zliczać jedynie do 65535. W tym przypadku można uzyskać rozszerzenie zakresu pomiarowego przez zwiększenie czasu obiegu trzeciej pętli, np. za pomocą kilku pustych rozkazów lub pętli opóźnienia. Trzeba oczywiście odpowiednio skorygować stałą w 320-tej linii programu głównego (Basic).

2.2.4. Pomiar okresu drgań

Do pomiaru okresu drgań są stosowane dwie metody, przy czym różnice między nimi ograniczają się do części układowej.

Parametry

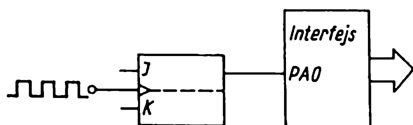
- maksymalna wartość okresu mierzonego, w zależności od metody: 1 lub 2 s,
- dokładność 0,2% przy 100 ms, 2% przy 1 ms.

Sprzęt (układ pomiarowy)

Część układowa urządzenia, niezależnie od metody, jest bardzo prosta. Jeśli współczynnik wypełnienia przebiegu jest równy dokładnie 0,5 (stosunek czasu trwania impulsu do czasu przerwy jak 1:1), to mierzy się po prostu — omówioną

poprzednio metodą — szerokość impulsów i mnoży przez dwa. A więc w linii 320 programu (Basic) wystarczy 2-krotnie zmniejszyć stałą (rys. 2.43). Jako układu pomiarowego używa się także układu z rys. 2.40.

Przy dowolnym współczynniku wypełnienia znajduje zastosowanie nieznacznie rozbudowany układ, pokazany na rys. 2.44. Włączony na wejście układu przerzutnik zmienia stan zawsze podczas narastającego albo tylko opadającego zbocza sygnału. Jego wyjście pozostaje więc w stanie wysokim przez czas odpowiadający okresowi przebiegu. Czas ten mierzy się następnie za pomocą programu (rys. 2.43) opisanego w poprzednim punkcie, przy czym nie ma tu potrzeby ingerować w program. Ulepszenia zaproponowane w p. 2.2.3 stosują się również do tej metody.



Rys. 2.44. Układ do pomiaru okresu. Użyty przerzutnik może być dowolnym przerzutnikiem typu J-K wyzwalanym zboczem

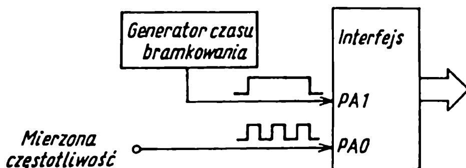
2.2.5. Pomiar częstotliwości

Parametry

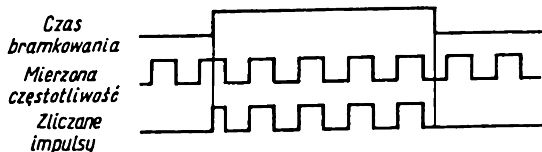
- pomiar częstotliwości: od ok. 2 do 9000 Hz,
- dokładność przy czasie pomiaru 0,5 s: 0,05% przy 9000 Hz, 15% przy 2 Hz.

Sprzęt (układ pomiarowy)

Budowa sprzętu jest tu szczególnie prosta, ponieważ właściwy pomiar jest realizowany za pomocą programu. Sposób dołączenia interfejsu pokazano na rys. 2.45, a zasadę pomiaru — na rys. 2.46. Generator czasu pomiaru —



Rys. 2.45. Układ do pomiaru częstotliwości



Rys. 2.46. Zasada pomiaru częstotliwości

najczęściej, ze względu na wymaganą stabilność częstotliwości, generator kwarcowy — wytwarza impuls prostokątny określający czas pomiaru. Czas ten ogranicza od dołu zakres mierzonych częstotliwości, ponieważ przynajmniej jeden impuls mierzonego sygnału powinien zostać zliczony. Oznacza to, że: im dłuższy jest czas pomiaru, tym dokładniejsze wskazanie, tym mniejsza częstotliwość graniczna i wolniejszy przebieg pomiaru. W najczęściej spotykanych miernikach częstotliwości czasy te wybiera się zwykle ze zbioru 1 s; 0,1 s itd. W przypadku komputera wybór jest w zasadzie dowolny dzięki jego możliwościom obliczeniowym. Jeśli np. wybierzemy czas 0,5 s, to liczbę zarejestrowanych impulsów trzeba będzie pomnożyć przez dwa, aby otrzymać częstotliwość w Hz.

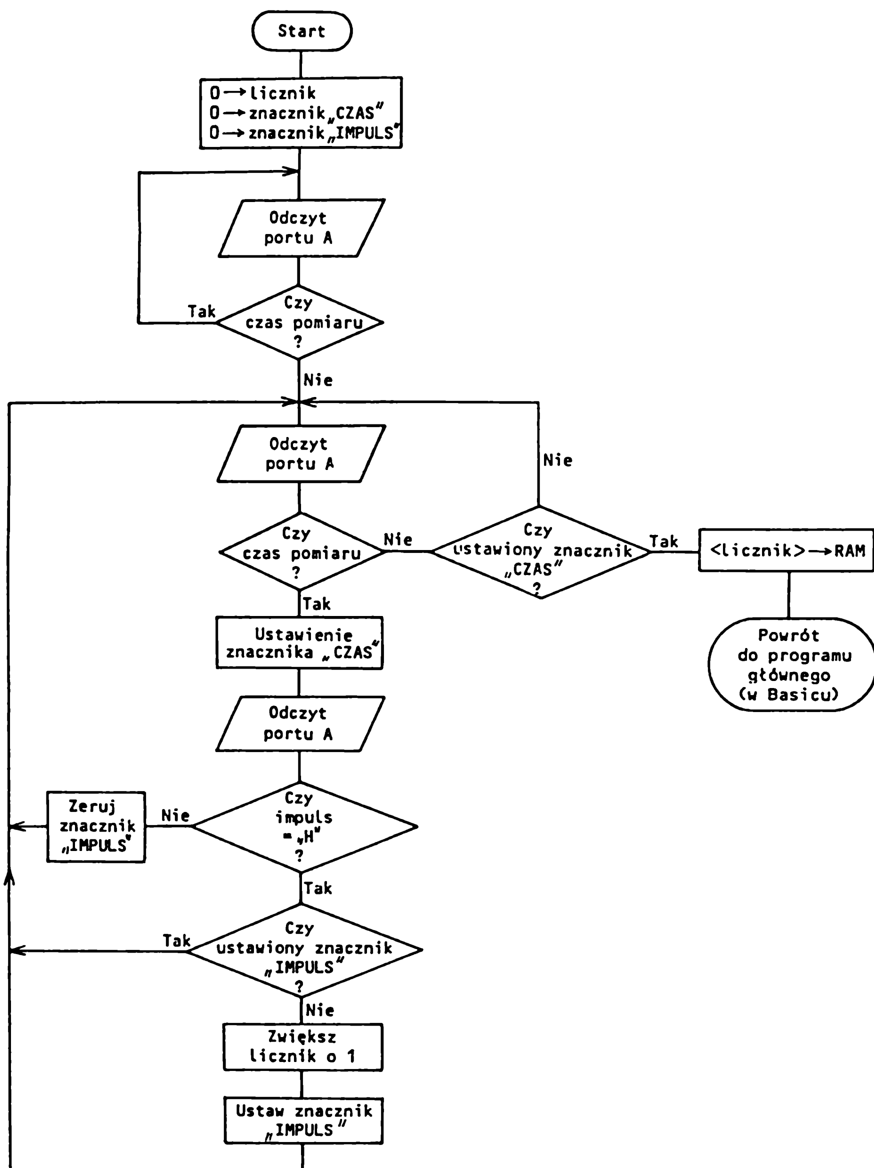
Program

Program do pomiaru częstotliwości składa się z dwóch części: z podprocedury asemblerowej, która jest właściwą procedurą pomiarową, oraz z części napisanej w Basicu, której zadaniem jest przetworzenie danej pomiarowej i wysłanie na ekran monitora.

Na rysunku 2.47 jest pokazana ogólna sieć działań asemblerowej części programu. Aby przekonać Czytelnika do pewnej techniki programowania stosowanej często w zagadnieniach sterowania i regulacji, zostanie tutaj przedstawiony elegancki program, choć — być może ze względu na swoją złożoność — trudniejszy do zrozumienia. Żeby pokazać zalety tej — polegającej na wprowadzeniu znaczników — techniki, prześledźmy raz jeszcze sieć działań programu do pomiaru szerokości impulsu (rys. 2.41). Istotą wymienionej procedury są trzy kolejne pętle, przy czym przejście do następnej z nich wymaga spełnienia określonego warunku. Jeśli warunek ten zostanie spełniony dopiero po dłuższym okresie czasu, jak to ma miejsce powiedzmy w przypadku kontroli temperatury w pralce, to komputer przez cały ten czas będzie obsługiwać tę pętlę bez możliwości obsługi innych procesów, jak np. sprawdzanie, czy jest już wymagana ilość wody.

Jeżeli ma być obsługiwana większa liczba procesów, np. pomiaru i/lub sterowania, to taki tryb pracy jest nie do przyjęcia. Referent, który ma porozumieć się z wieloma klientami, jeśli nie będzie mógł dodzwonić się do pierwszego z nich, to nie będzie przecież czekał dotąd, aż tamten podniesie słuchawkę. Zamiast tego, po krótkim sprawdzeniu zrobi sobie na kartce odpowiednią adnotację — znacznik — i natychmiast zabierze się do wybierania następnego numeru. W ten sposób może — poświęcając minimum czasu na czekanie — osiągnąć maksymalny efekt bez obawy, że o którymś z klientów zapomni.

Ten organizacyjny środek pomocniczy wykorzystuje się także w przypadku złożonych programów techniki pomiarowej, sterowania i regulacji, wprowadzając znaczniki. Taki znacznik może być pewną określoną liczbą (np. 0 lub 1) w rejestrze, ewentualnie pojedynczym bitem w bajcie znacznikowym i wówczas może być selektywnie ustawiony lub sprawdzany za pomocą procedury



Rys. 2.47. Ogólna sieć działań programu "POMIAR CZĘSTOTLIWOŚCI" (w asemblerze)

maskowania. Ostatnia z metod jest użyteczna wówczas, gdy potrzebnych jest wiele znaczników, a nie ma do dyspozycji wystarczającej liczby rejestrów.

W programie "POMIAR CZĘSTOTLIWOŚCI" zastosowano dwa znaczniki: znacznik "CZAS" pomiaru i znacznik "IMPULS". Pierwszy z nich jest zapamiętywany w rejestrze B, a drugi w rejestrze C. Para rejestrów HL służy jako licznik rejestrowanych w czasie pomiaru (czasie bramkowania) impulsów. Na początku programu (asembler) znaczniki i licznik zostają wyzerowane. Ponieważ procedura wywoływana jest przez program główny (Basic) w przypadkowym momencie, trzeba zabezpieczyć się przed sytuacją, w której odliczanie rozpocznie się wewnątrz czasu bramkowania powodując fałszywe wskazanie. Z tej racji komputer pozostanie w pętli aż do zakończenia czasu bramkowania. Teraz dopiero rozpoczyna się właściwy pomiar, w trakcie którego muszą być stale selektywnie obrabiane dwa, w pełni niezależne od siebie sygnały.

Po odczycie informacji z portu A następuje sprawdzenie, czy rozpoczął się czas bramkowania (stan wysoki na wejściu PA1). Załóżmy dla łatwiejszego zrozumienia procedury, że tak jest i że wcześniej ustawiony został znacznik "CZAS". Ponieważ podczas maskowanego sprawdzania odczytanej informacji zawartość akumulatora została zmieniona, port A musi być ponownie odczytany. Teraz następuje sprawdzenie, czy jest impuls na PA0 (PA0 = 1). Jeśli tak, i jeśli to było pierwsze sprawdzenie, zostanie zwiększony o jeden stan licznika (rejestry HL) i natychmiast ustawiony będzie znacznik "IMPULS".

Przy następnym przebiegu programu (obiegu pętli) zarówno wejście czasu bramkowania, jak i wejście sygnału mierzonego (PA1 i PA0) będą nadal w stanie wysokim. Ponieważ znacznik "IMPULS" jest teraz ustawiony (= 1), impuls nie zostanie po raz drugi policzony. Jeśli w którymś z następnych obiegów pętli impuls się skończy, to znacznik "IMPULS" zostanie wyzerowany, a przez to licznik będzie znowu gotowy do policzenia następnego impulsu. Gdy w trakcie obiegania pętli zakończy się czas bramkowania, komputer rozpozna to, badając stan wejścia PA1 (pojawi się na nim "0") oraz stan znacznika "CZAS" (znacznik będzie ustawiony). W tym przypadku zmierzona liczba impulsów przepisana zostanie z rejestrów HL do pamięci RAM i procedura (asembler) zakończy się powrotem do programu głównego.

Na rysunku 2.48 pokazano procedurę napisaną w asemblerze łącznie z kodami maszynowymi, a na rys. 2.49 napisaną w Basicu procedurę programu "POMIAR CZĘSTOTLIWOŚCI". Ta ostatnia składa się z trzech części: w pierwszej z nich jest przygotowany ekran monitora, w drugiej — po zaprogramowaniu interfejsu (linia 200) i ustawieniu adresu początkowego programu maszynowego — następuje wprowadzenie kodów tego programu do pamięci RAM za pomocą linii DATA (linia 260), poczynając od adresu 32512. W trzeciej wywołana zostaje procedura pomiarowa (linia 280), a następnie odczytana z pamięci RAM zmierzona liczba impulsów zostanie przeliczona na częstotliwość i wysłana na ekran.

Adres (hex.)	Mnemonika	Kody masz. (hex.)	Kody masz. (dzies.)	Objasnienia
7F00	DI	F3	243	Maska przerwan
01	LXI HL,0000	21	33	Zerowanie licznika
02		00	0	
03		00	0	
04	MVI B,00	06	6	Zerowanie znacznika
05		00	0	CZAS
06	MVI C,00	0E	14	Zerowanie znacznika
07		00	0	IMPULS
08	IN 04	DB	219	Odczyt portu A
09		04	4	
0A	ANI 02	E6	230	Maskowanie bitu 1
0B		02	2	(AND)
0C	JNZ 7F08	C2	194	Bit 1 = 1 ?
0D		08	9	
0E		7F	127	
0F	IN 04	DB	219	Odczyt portu A
10		04	4	
11	ANI 02	E6	230	Maskowanie bitu 1
12		02	2	
13	JZ 7F30	CA	202	Bit 1 = 0 ?
14		30	48	
15		7F	127	
16	MVI B,01	06	6	Ustawienie znacz.
17		01	1	CZAS
18	IN 04	DB	219	Odczt portu A
19		04	4	
1A	ANI 01	E6	230	Maskowanie bitu 0
1B		01	1	
1C	JZ 7F2B	CA	202	Bit 0 = 0 ?
1D		2B	43	
1E		7F	127	
1F	MOV A,C	79	121	Znacznik IMPULS
20	ANI 01	E6	230	do Akumulatora
21		01	1	i maskowanie
22	JNZ 7F0F	C2	194	Znacznik IMPULS
23		0F	15	ustawiony ?
24		7F	127	
25	INX HL	23	35	Zwieksz licznik o 1
26	MVI C,01	0E	14	Ustawienie znacz.
27		01	1	IMPULS
28	JMP 7F0F	C3	195	
29		0F	15	
2A		7F	127	
2B	MVI C,00	0E	14	Zerowanie znacznika
2C		00	0	IMPULS
2D	JMP 7F0F	C3	195	
2E		0F	15	
2F		7F	127	
30	MOV A,B	78	120	Znacznik CZAS do
31	ANI 01	E6	230	Akumulatora i mas-
32		01	1	kowanie
33	JZ 7F0F	CA	202	Znacznik CZAS wyze-
34		0F	15	rowany ?
35		7F	127	
36	SHLD 7F80	22	34	Zapamietanie HL
37		80	128	(stan licz.) w pam
38		7F	127	o adr. 7F80 i 7F81
39	EI	FB	251	Zdjecie maski przer.
3A	RET	C9	201	Powrot do programu glownego (BASIC)

Rys. 2.48. Program "POMIAR CZĘSTOTLIWOŚCI". Procedura napisana w języku asemblera oraz odpowiednie kody maszynowe

```
100 **** PROGRAM :"POMIAR CZESTOTLIWOSCI" ****
110 **** NAZWA PROG.:"CZESTOT"            ****
120 CLS
130 REM LINIE 140-190 : RAMKA NA EKRAN
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) SET (127,Y)
190 NEXT Y
200 OUT 7,153 : REM ZAPROGRAMOWANIE INTERFEJSU
210 DEFUSR 0 = &H7F00
220 FOR X = 32512 TO 32570
230 READ D
240 POKE X,D
250 NEXT X
260 DATA 243,33,0,0,6,0,14,0,219,4,230,2,194,8,127,219,4,230,2,
    202,48,127,6,1,219,4,230,1,202,43,127,121,230,1,194,15,127,
    35,14,1,195,15,127,14,0,195,15,127,120,230,1,202,15,127,34,
    128,127,251,201
270 PRINT@ 328;"P O M I A R  C Z E S T O T L I W O S C I"
280 S = USR0(X)
290 A = PEEK(32640)
300 B = PEEK(32641)
310 F = 256 * B + A
320 PRINT @648,"CZESTOTLIWOSC WYNOSI : ";
330 M$ ="####"
340 PRINT @715, USING M$; F ;
350 PRINT " HZ"
360 GOTO 280
370 END
```

Rys. 2.49. Program "POMIAR CZĘSTOTLIWOŚCI" — część napisana w języku Basic

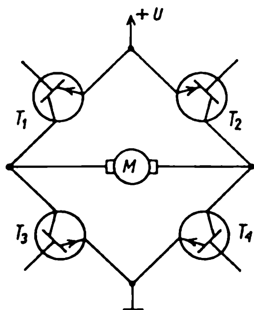
Dokładność takiego pomiaru częstotliwości zależy w zasadzie od dokładności czasu bramkowania. Jeśli to możliwe, powinien tu być użyty układ z generatorem kwarcowym. Górna częstotliwość graniczna jest określona czasem przebiegu programu i nie można jej zwiększyć, chyba że na wejściu zestawu pomiarowego zostanie włączony dzielnik częstotliwości (przerzutnik albo licznik). Dolną częstotliwość można obniżyć wybierając dłuższy czas pomiaru. W przypadku małych częstotliwości, ze względu na wydłużony czas pomiaru należy mierzyć okres i częsotliwość obliczać za pomocą programu (Basic) ze wzoru $f = 1/T$. Zapewnia to znacznie większą dokładność pomiaru.

3. Sterowanie za pomocą komputera wyposażonego w Basic

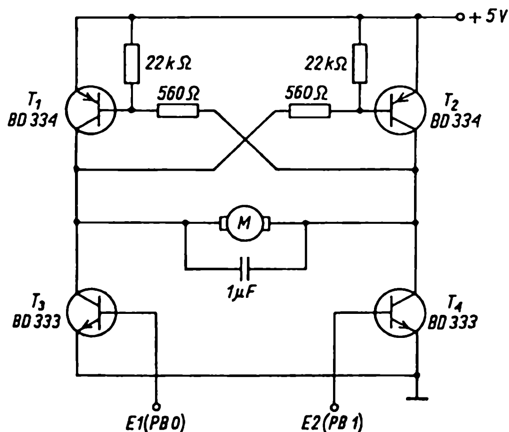
3.1. Sterowanie silnikiem prądu stałego

Parametry

- trzy stany pracy silnika: obroty w lewo, obroty w prawo, stan spoczynku;
- sterowanie bezpośrednie: przez wprowadzenie parametru (*X* — obroty w lewo, *Y* — w prawo, *Z* — stop);
- praca w czasie rzeczywistym: programowanie dziewięciu cykli pracy silnika, przy czym dla każdego cyklu moment startu, zatrzymania oraz kierunek obrotu można wybrać dowolnie;
- nominalne napięcie pracy silnika: 5 V, maksymalny pobór prądu: 3 A.



Rys. 3.1. Zasada sterowania silnikiem prądu stałego



Rys. 3.2. Schemat ideowy układu sterowania silnikiem prądu stałego

Sprzęt (układ sterowania silnikiem)

Na rysunku 3.1 przedstawiono zasadę działania układu sterowania silnikiem. Cztery tranzystory tworzą tu układ mostka, w którego przekątną jest włączony silnik. Dla silnika prądu stałego, którego kierunek obrotów jest określony kierunkiem przepływu prądu, odpowiednie dwa tranzystory muszą być nasycone, a dwa pozostałe — zatkane. Dla jednego kierunku obrotów będą więc nasycone T_1 i T_4 (T_2, T_3 — zatkane), a dla przeciwnego — T_2 i T_3. Zatrzymanie silnika uzyskuje się przez zatkanie wszystkich lub przynajmniej dwóch dolnych tranzystorów.

Na rysunku 3.2 pokazano schemat ideowy układu. Aby uniknąć oddzielnego sterowania wszystkich czterech tranzystorów, zastosowano tu pewien trik układowy. Tranzystory T_3 i T_4 wysterowują odpowiednio tranzystory T_2 i T_1. Takie sprzężenie "na krzyż" powoduje, że układ działa podobnie jak przerzutnik, przy czym otrzymuje się następującą tablicę sterowania:

Sygnały wejściowe E1	E2	Działanie
L	L	Stop
L	H	Obroty w prawo
H	L	Obroty w lewo
H	H	Stan niedozwolony (zwarcie zasilania)

Ponieważ pojedyncze tranzystory mocy obciążałyby zbytnio wyjścia portu (ze względu na zbyt duży prąd bazy), użyto układów Darlingtona, które mają duży współczynnik wzmocnienia prądowego.

Program

Większości czytelników jest na pewno znany sposób przedstawienia działania programu za pomocą *sieci działań* (patrz przykłady z poprzedniego rozdziału). Metoda ta, w której używa się zaledwie kilku standardowych symboli, ilustruje w łatwej do zrozumienia formie logiczne następstwo poszczególnych operacji wykonywanych przez komputer (następstwo akcji). Niestety, w przypadku rozrastających się programów, sieci działań szybko stają się mało przejrzyste, zwłaszcza wtedy, kiedy zajmują wiele stron. Stąd też dla dużych programów są one coraz rzadziej stosowane, a w najlepszym razie przyjmują postać ogólnych schematów blokowych, w których każdej "klatce" odpowiada kilka lub kilkanaście instrukcji bądź też cały moduł programu. Inną formą przedstawienia logicznego następstwa kroków programu są *diagramy syntaktyczne* lub użycie *pseudokodów*. Ten ostatni sposób zostanie tu pokazany (dla przypadku sterowania silnikiem) jako alternatywny w stosunku do metody sieci działań.

Chodzi tu o przejrzystą, problemowo zorientowaną, strukturalną formę przedstawienia, w której następstwo poszczególnych kroków programu zostałoby opisane za pomocą łatwych do zrozumienia tekstów. Znajdują przy tym zastosowanie pewne określone słowa, tzw. pseudokody: "begin" (początek), "end" (koniec), "if" (jeśli), "then" (to, wówczas), "else" (w przeciwnym razie), "until" (dopóki), "do" (wykonuj). Konwersja procedury napisanej w takiej postaci na program w języku Basic lub Pascal nie sprawia na ogół trudności. Przykład przedstawiony poniżej, również powinien być dla czytelnika łatwo zrozumiały.

```
STEROWANIE SILNIKIEM
BEGIN
   przygotowanie miejsca w pamięci
   zaprogramowanie interfejsu
   UNTIL nie "Break"
       DO wysłanie (na ekran) 'wybierz'
          wysłanie (na ekran) 'tryb sterowania
                                   A) bezpośrednie, B) w czasie rzeczywistym'
          wprowadzenie (z klawiatury) wybranego trybu sterowania
              IF wybór = A
              THEN [STEROWANIE BEZPOŚREDNIE]
              IF wybór = B
              THEN [CZAS RZECZYWISTY]
END  STEROWANIE SILNIKIEM
```

Wyrażenia w nawiasach kwadratowych zostaną przedstawione w postaci osobnego opisu. Na ekranie pojawią się teksty oznaczone apostrofem.

```
STEROWANIE BEZPOŚREDNIE
BEGIN
wygaszenie ekranu
DO   wysłanie 'sterowanie silnikiem — klawisze X, Y, Z'
     wysłanie 'X = obroty w lewo'
     wysłanie 'Y = obroty w prawo'
     wysłanie 'Z = zatrzymanie'
     wysłanie 'A = wyjście z procedury ster. bezpośr.'
     wprowadzenie znaku z klawiatury
     UNTIL klawisz ≠ A
     DO     IF      klawisz = X
            THEN    wys łanie sygnału sterującego obroty w lewo
            IF      klawisz = Y
            THEN    wysłanie sygnału sterującego obroty w prawo
```

```
            IF    klawisz = Z
            THEN wysłanie sygnału sterującego zatrzymanie
            wprowadzenie znaku z klawiatury
END  STEROWANIE BEZPOŚREDNIE
```

CZAS RZECZYWISTY

```
BEGIN
wygaszenie ekranu
DO    wysłanie 'możesz zaprogramować maksimum 9 cykli pracy'
      wysłanie 'wprowadź następujące dane'
      wysłanie '1. czas włączenia'
      wysłanie '2. czas wyłączenia'
      wysłanie '3. kierunek obrotów'
      wprowadzenie [DANE]
FOR   licznik 1 = 1 TO licznik
DO    IF      czas włączenia (licznik 1) > = czas wyłączenia (licznik 1)
      THEN    wysłanie komunikatu — błąd
              wprowadzenie [DANE (licznik 1)]
      IF      czas włączenia (licznik 1+1) < czas wyłączenia (licznik 1)
      THEN    wysłanie komunikatu — błąd
              wprowadzenie [DANE (licznik 1+1)]
FOR   Licznik 2 = 1 TO licznik
DO    UNTIL   czas włączenia (licznik 2) > czas rzeczywisty
      DO      oczekiwanie
              wysłanie dane (licznik 2)
      IF      kierunek obrotów (licznik 2) = w lewo
      THEN    wysłanie sygnału sterującego obroty w lewo
      ELSE    wysłanie sygnału sterującego obroty w prawo
      UNTIL   czas wyłączenia (licznik 2) > czas rzeczywisty
      DO      oczekiwanie
      wysłanie sygnału sterującego zatrzymanie
      wysłanie [LISTA]
END CZAS RZECZYWISTY
```

DANE

```
BEGIN
REPEAT licznik = licznik + 1
         REPEAT  wprowadzenie czasu włączenia (licznik)
         TO      czas włączenia (licznik) > = 00:00:00 i ≤23:59:59
         wysłanie   [LISTA]
```

```
        REPEAT  wprowadzenie czasu wyłączenia (licznik)
        TO      czas wyłączenia (licznik) > = 00:00:00 i < 23:59:59
        wysłanie  [LISTA]
        wprowadzenie kierunku obrotów (licznik)
        wysłanie  [LISTA]
        wysłanie 'czy chcesz zaprogramować następne cykle?'
        wprowadzenie odpowiedzi
UNTIL odpowiedź = nie lub licznik > 9
END DANE
```

LISTA

```
BEGIN
wygaszenie ekranu
FOR  licznik 3 = 1 TO licznik
DO   wysłanie  licznik 3, czas włączenia (licznik 3)
     wysłanie  licznik 3, czas wyłączenia (licznik 3)
     wysłanie  kierunek obrotów (licznik 3)
END  LISTA
```

Na rysunku 3.3 przedstawiono program sterowania silnikiem napisany w języku Basic. Ponieważ podany został dokładny opis za pomocą pseudokodów, program ten nie wymaga w zasadzie szczegółowych objaśnień. Warto jednak zwrócić uwagę na zabezpieczenie programowe przed wprowadzeniem błędnych danych (program może służyć jako typowy przykład). Program ten rozpoznaje wiele różnych typów błędów, między innymi:

— deklarację czasu < 0 lub ⩾ 24 godz,
— czas włączania ⩾ czasu zatrzymania,
— zachodzenie na siebie przedziałów czasowych (cykli).

Na tym przykładzie można się jednocześnie przekonać, że wprowadzenie zabezpieczeń prowadzi do bardzo rozbudowanych programów. Trzeba więc niekiedy rozstrzygnąć, czy pożądany skutek w postaci zapobieżenia szkodom powstałym w wyniku błędnej obsługi usprawiedliwia dodatkowy wysiłek włożony przez programistę.

Ulepszenia

Przedstawiony na rys. 3.2 układ może być użyty również w przypadku silników większej mocy. Jednak tranzystory Darlingtona trzeba wtedy zastąpić innymi, o większym maksymalnym prądzie kolektora i większym wzmocnieniu prądowym. Może też być konieczna zmiana wartości rezystorów sterujących bazy tranzystorów T_1 i T_2.

Rys. 3.3. Program (Basic) "STEROWANIE SILNIKIEM PRĄDU STAŁEGO" (s. 79—84)

```
1000 REM  **** PROGRAM STEROWANIA SILNIKIEM        ****
1010 REM NAZWA PROGRAMU : "MOTSTER"                ****
1020 CLEAR 1000
1030 OUT 7,144
1040 OUT 5,0
1050 CLS
1060 PRINT ""
1070 PRINT "                 S T E R O W A N I E   S I L N I K A    "
1080 PRINT "":PRINT ""
1090 PRINT  TAB(22) "WYBIERZ TYP STEROWANIA :"
1100 PRINT ""
1110 PRINT ""
1120 PRINT " A)  STEROWANIE BEZPOSREDNIE"
1130 PRINT " B)  STEROWANIE PROGRAMOWANE"
1140 A$ = INKEY$
1150 IF A$ = "A" GOTO 1180
1160 IF A$ = "B" GOTO 1390
1170 GOTO 1140
1180 CLS
1190 B = 31
1200 PRINT "":PRINT ""
1210 PRINT TAB(20) "STEROWANIE BEZPOSREDNIE"
1220 PRINT "":PRINT ""
1230 PRINT "SILNIK STEROWANY JEST KLAWISZAMI   :  X  ,  Y  ,  Z"
1240 PRINT "":PRINT ""
1250 PRINT " X - OBROTY W LEWO"
1260 PRINT " Y - OBROTY W PRAWO"
1270 PRINT " Z - STOP"
1280 PRINT "":PRINT ""
1290 PRINT "PRZYCISNIECIE KLAWISZA 'A' - PRZERWANIE STEROWANIA BEZPOSREDNIEGO"
1300 A$ = INKEY$
1310 IF A$ = "X" THEN  OUT 5,1 : B = 25
1320 IF A$ = "Y" THEN  OUT 5,2 : B = 28
1330 IF A$ = "Z" THEN  OUT 5,0 : B = 31
1340 IF A$ = "A" THEN  OUT 5,0 : GOTO 1050
```

```
1350 SET (35,B)
1360 FOR C = 0 TO 50 : NEXT
1370 RESET (35,B)
1380 FOR C = 0 TO 50 : NEXT : GOTO 1300
1390 CLS
1400 PRINT "":PRINT ""
1410 PRINT TAB(19)  "STEROWANIE PROGRAMOWANE"
1420 PRINT ""
1430 PRINT "MOZNA ZAPROGRAMOWAC DO 9 CYKLI PRACY"
1440 PRINT ""
1450 PRINT "PROSZE PODAC NASTEPUJACE DANE  :"
1460 PRINT ""
1470 PRINT "1. CZAS WLACZENIA SILNIKA"
1480 PRINT "2. CZAS WYLACZENIA SILNIKA"
1490 PRINT "3. KIERUNEK OBROTOW"
1500 Z = Z + 1
1510 C$ = ":"
1520 PRINT ""
1530 PRINT "PROSZE PODAC GODZINE ROZPOCZECIA"
1540 PRINT Z;".  CYKLU PRACY"
1550 GOSUB 2850
1560 IF D3$ > "23" GOTO 1580
1570 IF D3$ >= "00" GOTO 1590
1580 PRINT "BLEDNA WARTOSC" : GOTO 1530
1590 E$(Z) = D3$ + C$
1600 CLS
1610 GOSUB 2990
1620 PRINT ""
1630 PRINT "PROSZE PODAC MINUTY"
1640 GOSUB 2850
1650 IF D3$ > "59" GOTO 1670
1660 IF D3$ >= "00" GOTO 1680
1670 PRINT "BLEDNA WARTOSC !!" : GOTO 1630
1680 E$(Z) = E$(Z) + D3$ + C$
1690 CLS
```

```
1700 GOSUB 2990
1710 PRINT "PROSZE PODAC SEKUNDY"
1720 GOSUB 2850
1730 IF D3$ > "59" GOTO 1750
1740 IF D3$ >= "00" GOTO 1760
1750 PRINT "BLEDNA WARTOSC !!" : GOTO 1710
1760 E$(Z) = E$(Z) + D3$
1770 CLS
1780 GOSUB 2990
1790 PRINT ""
1800 PRINT "PROSZE PODAC GODZINE WYLACZENIA DLA"
1810 PRINT Z;".  CYKLU PRACY"
1820 GOSUB 2850
1830 IF D3$ > "23" GOTO 1850
1840 IF D3$ > "00" GOTO 1860
1850 PRINT "BLEDNA WARTOSC  !!" : GOTO 1800
1860 E1$(Z) = D3$ + C$
1870 CLS
1880 GOSUB 2990
1890 PRINT "PROSZE PODAC MINUTY"
1900 GOSUB 2850
1910 IF D3$ > "59" GOTO 1930
1920 IF D3$ >= "00" GOTO 1940
1930 PRINT "BLEDNA WARTOSC  !!" : GOTO 1890
1940 E1$(Z) = E1$(Z) + D3$ + C$
1950 GOSUB 2990
1960 PRINT ""
1970 PRINT "PROSZE PODAC SEKUNDY"
1980 GOSUB 2850
1990 IF D3$ > "59" GOTO 2010
2000 IF D3$ >= "00" GOTO 2020
2010 PRINT "BLEDNA WARTOSC  !!" : GOTO 1970
2020 E1$(Z) = E1$(Z) + D3$
2030 GOSUB 2990
2040 PRINT "":PRINT ""
```

```
2050 PRINT "PROSZE PODAC KIERUNEK OBROTOW"
2060 PRINT "":PRINT ""
2070 PRINT " X - W LEWO"
2080 PRINT " Y - W PRAWO"
2090 L$(Z) = INKEY$
2100 IF L$(Z) = "X" GOTO 2130
2110 IF L$(Z)= "Y" GOTO 2130
2120 GOTO 2090
2130 GOSUB 2990
2140 PRINT ""
2150 PRINT "DANE WPROWADZONE PRAWIDLOWO ? (T/N)"
2160 N$=INKEY$
2170 IF N$="N" GOTO 1510
2180 IF N$="T" GOTO 2200
2190 GOTO 2160
2200 PRINT ""
2210 IF Z .= 9 GOTO 2280
2220 IF H=1 GOTO 2470
2230 PRINT "CZY BEDA PROGRAMOWANE DALSZE CYKLE PRACY ?"
2240 F$=INKEY$
2250 IF F$="T" GOTO 1500
2260 IF F$="N" GOTO 2280
2270 GOTO 2240
2280 I=Z
2290 E$(Z+1)= "24"
2300 FOR G=1 TO I
2310 IF E$(G) < E1$(G) GOTO 2390
2320 CLS
2330 GOSUB 2990
2340 PRINT ""
2350 PRINT G;"CZAS WLACZENIA JEST WIEKSZY LUB ROWNY CZSOWI WYLACZENIA  !"
2360 PRINT "PROSZE SKORYGOWAC  !!"
2370 H=1 : Z=G
2380 GOTO 1510
2390 IF E1$(G) <= E$(G+1) GOTO 2470
```

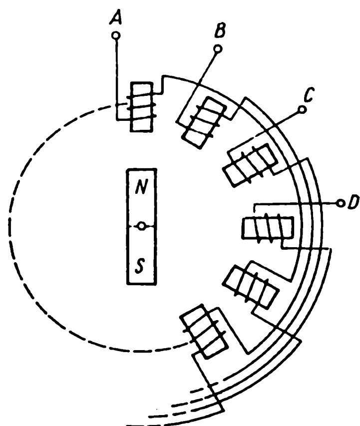
```
2400 CLS
2410 GOSUB 2990
2420 PRINT ""
2430 PRINT G+1;"CZAS WLACZENIA JEST MNIEJSZY NIZ";G;"CZAS WYLACZENIA  !"
2440 PRINT "PROSZE SKORYGOWAC  !!"
2450 Z=G+1 : H=1
2460 GOTO 1510
2470 NEXT
2480 IF H=1 THEN H=0 : GOTO 2300
2490 CLS
2500 GOSUB 2950
2510 FOR K=1 TO I
2520 T$ = RIGHT$(TIMES$,8)
2530 IF E$(K) > T$ GOTO 2520
2540 IF E1$(K) < T$ GOTO 2720
2550 IF L$(K) = "X" THEN OUT 5,1
2560 IF L$(K) = "Y" THEN OUT 5,2
2570 CLS
2580 PRINT ""
2590 PRINT "          ROZPOCZETY CYKL PRACY  :"
2600 PRINT @ 164,K;".  CZAS WLACZENIA  =";E$(K)
2610 PRINT @ 228,K;".  CZAS WYLACZENIA =";E1$(K)
2620 IF L$(K) = "X" THEN PRINT @ 292,"KIERUNEK OBROTOW - W LEWO "
2630 IF L$(K) = "Y" THEN PRINT @ 292,"KIERUNEK OBROTOW - W PRAWO"
2640 PRINT ""
2650 GOSUB 3010
2660 T$ = RIGHT$(TIMES$,8)
2670 IF E1$(K) > T$ GOTO 2660
2680 OUT 5,0
2690 CLS
2700 E$(K) = E$(K) + " *"
2710 GOSUB 2950
2720 NEXT
2730 Z=0
2740 FOR M=1 TO I
```

```
2750 E$(M) = ""
2760 E1$(M)= ""
2770 L$(M) = ""
2780 NEXT
2790 I=0
2800 O=0
2810 PRINT ""
2820 PRINT "CYKLE OZNACZONE  '* ' ZOSTALY ROZPOCZETE"
2830 FOR P=1 TO 8000 : NEXT
2840 GOTO 1050
2850 D1$ = INKEY$
2860 IF D1$ = "" GOTO 2850
2870 IF D1$ > "9" GOTO 2850
2880 IF D1$ < "0" GOTO 2850
2890 D2$ = INKEY$
2900 IF D2$ = "" GOTO 2890
2910 IF D2$ > "9" GOTO 2890
2920 IF D2$ < "0" GOTO 2890
2930 D3$ = D1$ + D2$
2940 RETURN
2950 PRINT "":PRINT "":PRINT ""
2960 PRINT "ZOSTALY ZAPROGRAMOWANE NASTEPUJACE CYKLE  :"
2970 PRINT ""
2980 GOTO 3010
2990 CLS
3000 PRINT "":PRINT ""
3010 O=2
3020 IF I <> 0 THEN O=I
3030 FOR L=1 TO O
3040 PRINT L;".  CZAS WLACZENIA ";E$(L);"     ";L;".  CZAS WYLACZENIA  ";E1$(L);
3050 IF L$(L) = "X" THEN PRINT "OBROTY - W LEWO"
3060 IF L$(L) = "Y" THEN PRINT "OBROTY - W PRAWO"
3070 IF L$(L) = "" THEN PRINT ""
3080 NEXT
3090 RETURN
```

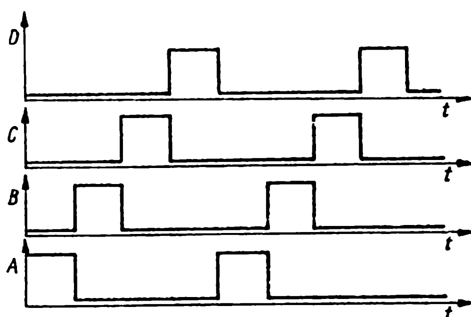
3.2. Sterowanie silnikiem krokowym

Silniki krokowe są stosowane wszędzie tam, gdzie wymagana jest dokładnie określona liczba obrotów, albo część obrotu wału silnika. Tak jest np. w przypadku mechanizmu stacji dyskowej, gdzie głowica zapisująco-odczytująca powinna zostać ustawiona dokładnie nad żądaną ścieżką albo, gdy głowica wiertarska sterowanej komputerem automatycznej obrabiarki musi przyjąć dokładnie zadaną pozycję.

Rysunek 3.4 ilustruje istotę konstrukcji silnika krokowego. Rotor w postaci magnesu stałego obraca się w kierunku oddziaływającego pola magnetycznego. Jeśli wysteruje się uzwojenia statora w ten sposób, że powstanie wirujące pole magnetyczne (rys. 3.5), to rotor będzie podążał za kierunkiem pola.



Rys. 3.4. Schemat budowy silnika krokowego



Rys. 3.5. Diagram czasowy impulsów prądu zasilającego uzwojenia silnika krokowego, przedstawionego na rys. 3.4

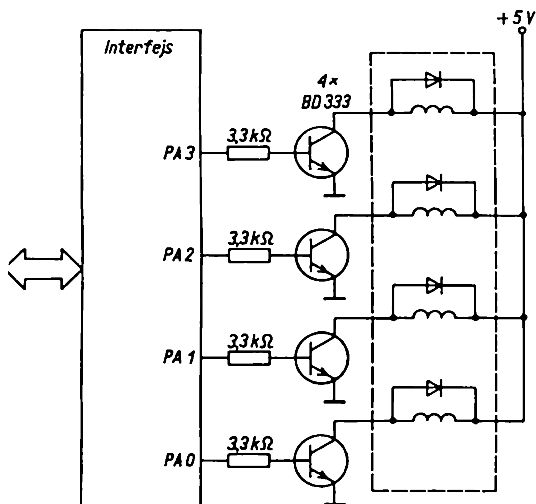
Przełączeniu prądu z jednego uzwojenia do następnego odpowiada obrót o pewien ściśle określony kąt. Całkowita liczba przełączeń wyznacza więc kąt obrotu twornika. Jeśli cykl przełączeń zostanie przerwany na określonym uzwojeniu, to rotor pozostanie w tej pozycji. Przy niezbyt dużej prędkości przełączania, rotor obraca się bez poślizgu (w stosunku do kierunku pola), z czego wynika, że jednostka sterująca nie musi otrzymywać informacji zwrotnej o chwilowym położeniu rotora, jak to jest wymagane w przypadku sterowania silnikiem prądu stałego. Maksymalna częstotliwość impulsów (kroków na sekundę) wynosi ok. 320 Hz, co daje maksymalną prędkość obrotową 8 obr/s.

Parametry

- obrót o zadany kąt,
- wykonanie zadanej liczby obrotów (w prawo lub w lewo),
- obrót o zadaną liczbę kwantów (kroków elementarnych).

Sprzęt (układ sterowania)

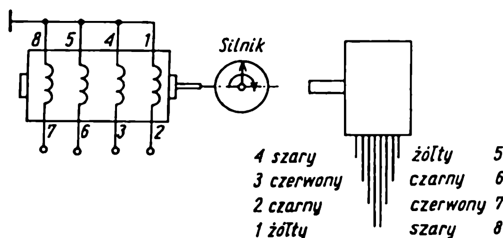
Na rysunku 3.6 pokazano układ sterowania silnikiem. Cztery użyte tranzystory Darlingtona zapewniają wymagane duże wzmocnienie prądowe, ponieważ wyjścia portu 8255 można obciążyć jedynie stosunkowo małym prądem. Diody włączone równolegle do uzwojeń silnika mają za zadanie zabezpieczać tranzystory przed przepięciami występującymi podczas wyłączania prądu.



Rys. 3.6. Schemat układu sterowania silnikiem krokowym (diody uniwersalne dowolnego typu)

Program

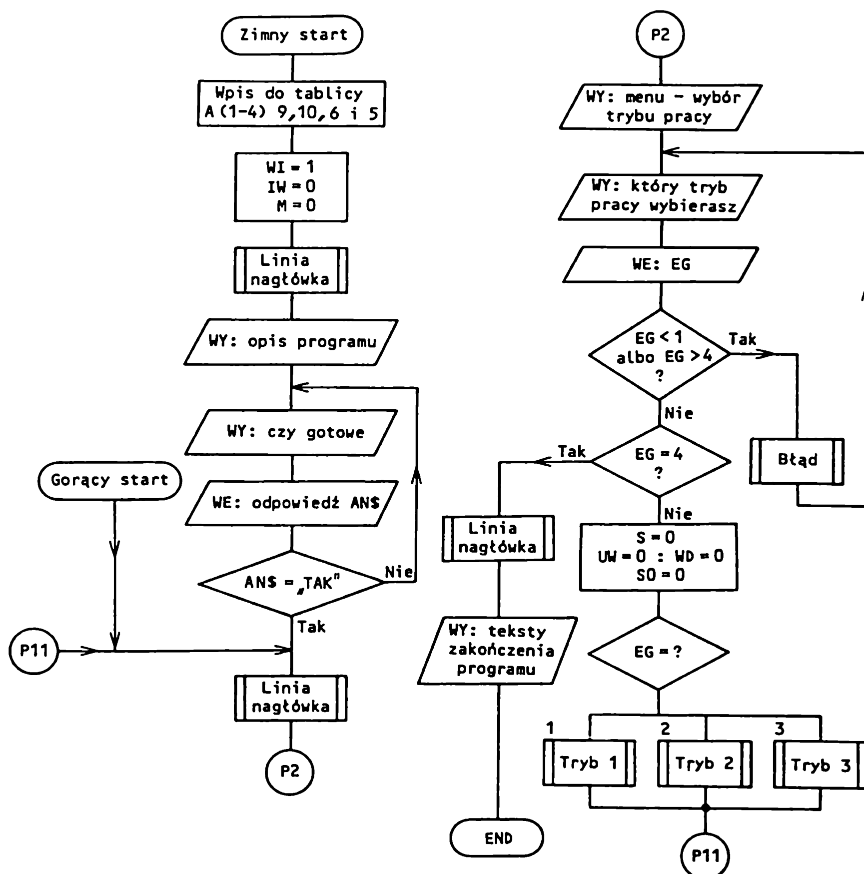
W urządzeniu zastosowano silnik krokowy firmy Valvo, o czterech uzwojeniach i napięciu pracy 5 V (rys. 3.7). Kolejność zasilania uzwojeń dla pojedynczego kąta



Rys. 3.7. Rozmieszczenie wyprowadzeń silnika krokowego firmy Valvo

obrotu i wymaganego kierunku obrotów podaje poniższa tablica (porównaj z danymi katalogowymi elementów firmy Valvo):

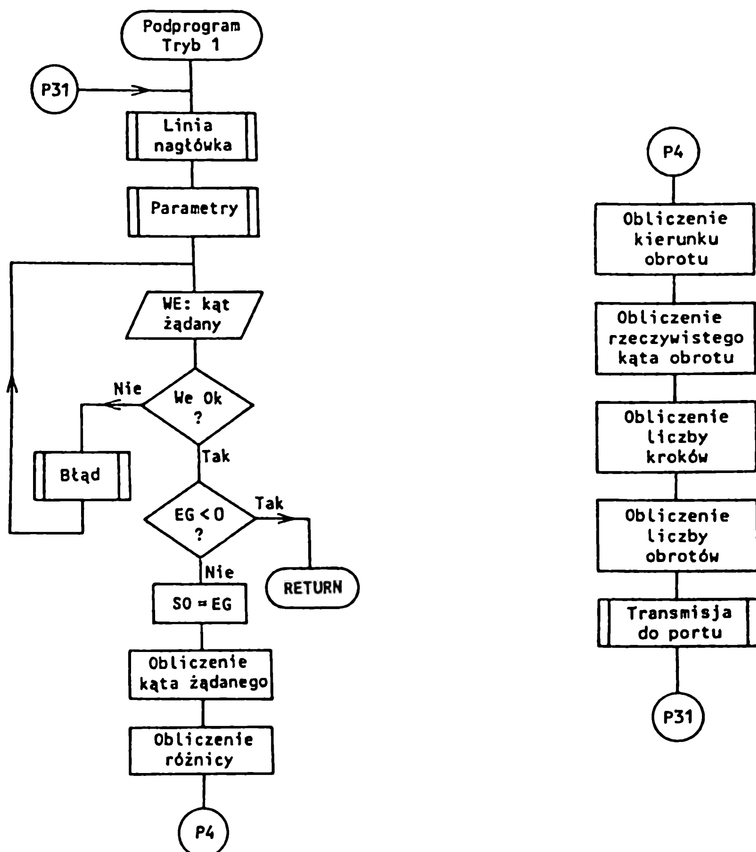
Kąt obrotu	Poziom na doprowadzeniach uzwojeń							
	obrót w prawo				obrót w lewo			
	7	6	3	2	7	6	3	2
0°	H	L	L	H	H	L	L	H
7°30′	L	H	L	H	H	L	H	L
15°	L	H	H	L	L	H	H	L
22°30′	H	L	H	L	H	L	H	L
30°	H	L	L	H	H	L	L	H



Rys. 3.8. Sieć działań programu "STEROWANIE SILNIKIEM KROKOWYM"

Zamieszczony układ kombinacji bitów, przekształcony na postać dziesiętną, daje dla jednego kierunku obrotów ciąg liczb: 9, 10, 6, 5. Po zmontowaniu układu sterowania i połączeniu z interfejsem komputera można sprawdzić poprawność jego działania za pomocą następującego prostego programu:

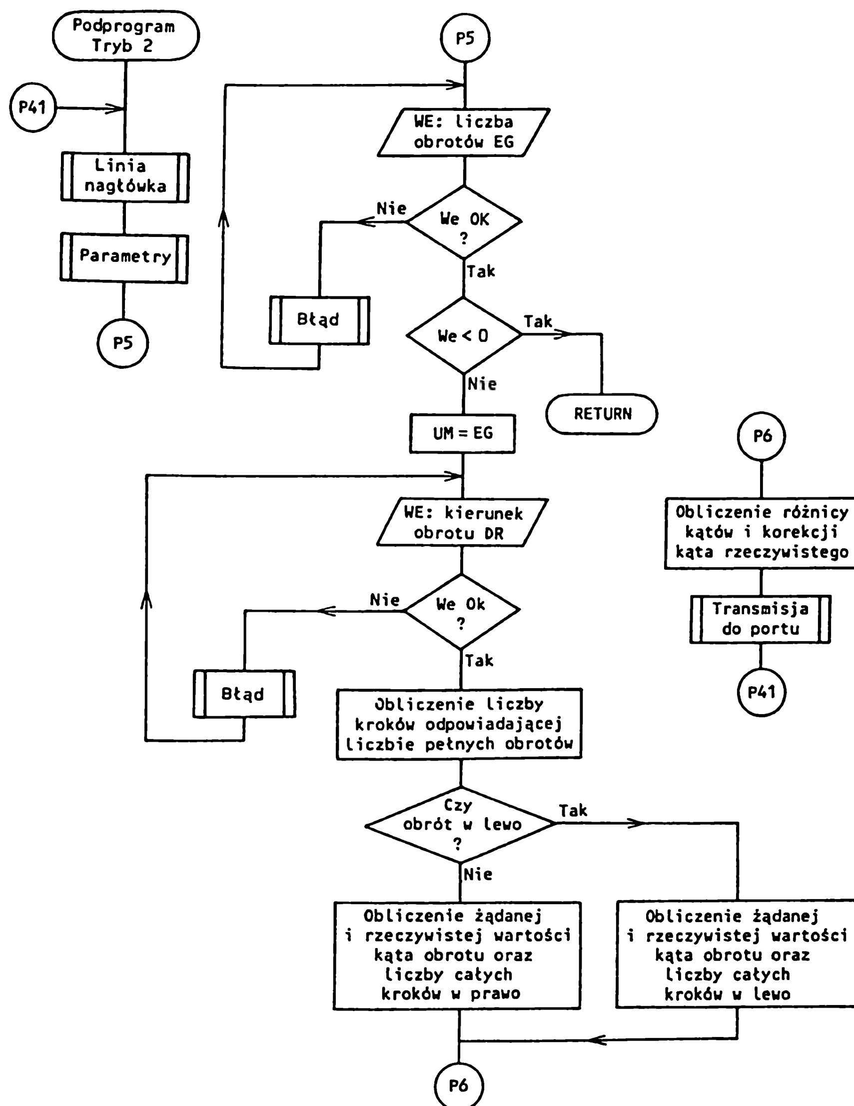
```
10  OUT 7,128   : REM ZAPROGRAMOWANIE INTERFEJSU
20  OUT 4,9     : REM WYKONANIE 1 KROKU
30  OUT 4,10    : REM WYKONANIE 2 KROKU
40  OUT 4,6     : REM WYKONANIE 3 KROKU
50  OUT 4,5     : REM WYKONANIE 4 KROKU
60  GOTO 20     : REM POWTARZANIE KROKÓW 1—4
```



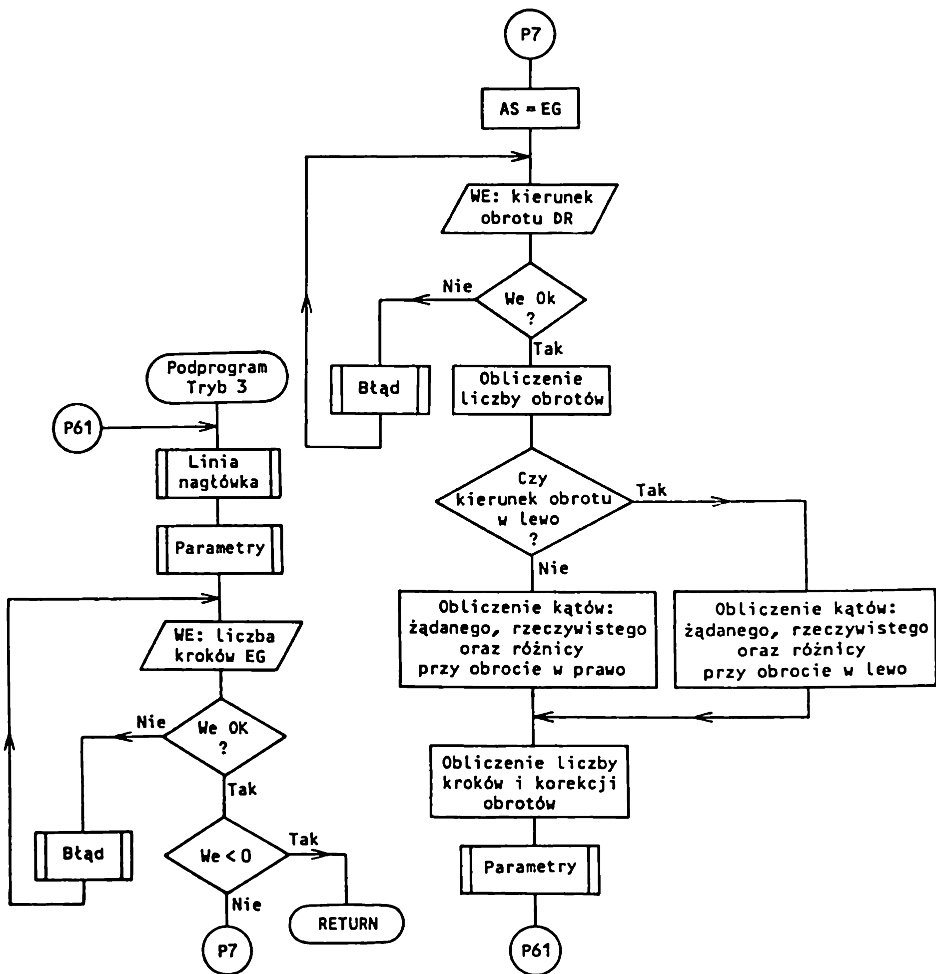
Rys. 3.9. Sieć działań podprogramu obsługi 1-go trybu pracy

REM (od ang. *remarks)* — czyli komentarze — służą tu nie tylko jako objaśnienia, ale są wręcz wymagane, aby ze względu na wyżej wzmiankowaną maksymalną prędkość obrotową spowolnić ciąg wysyłanych impulsów.

Sieci działań programu sterującego oraz niezbędnych podprocedur pokazano na rysunkach od 3.8 do 3.15, a na rys. 3.16 przedstawiono program w

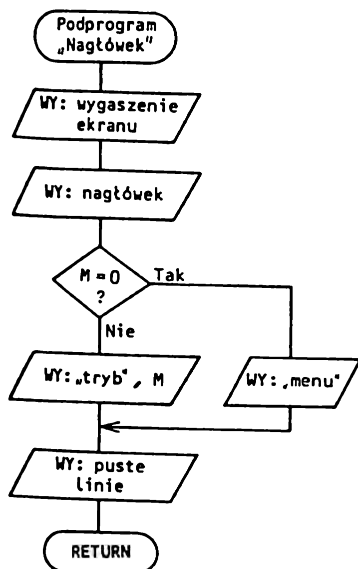


Rys. 3.10. Sieć działań podprogramu obsługi 2-go trybu



Rys. 3.11. Sieć działań podprogramu obsługi 3-go trybu

języku Basic. Jak widać, po nadaniu zmiennym wartości początkowych (inicjalizacja) oraz zaprogramowaniu interfejsu, na ekran monitora jest wysyłana informacja dla użytkownika. Między innymi poleca się użytkownikowi ustawić ręcznie rotor silnika w określonym położeniu wyjściowym (0 stopni). Ta część programu zostanie wykonana tylko raz, po starcie — stąd w sieci działań określenie "zimny start". Następnie zostanie podany spis "dań" — "menu", w celu wyboru trybu sterowania. "Menu" obejmuje wszystkie niezbędne wartości zmiennych i stosownie do tego użytkownik może zdecydować się na jeden z proponowanych trybów pracy. Procedura wysłania na ekran "menu" znajduje się w liniach 1190—1250. Do tej części programu można wykonać skok



Rys. 3.12. Sieć działań podprogramu wyświetlającego linie nagłówka

bezpośrednio po chwilowym przerwaniu wykonywania programu. Stąd w sieci działań określenie "gorący start".

UWAGI DO POSZCZEGÓLNYCH TRYBÓW PRACY:

Tryb 1: Ustawienie rotora w nowym, określonym położeniu. Komputer oczekuje od użytkownika podania położenia kątowego rotora silnika, jakie powinien przyjąć po wykonaniu obrotu. Decyzję, w którym kierunku powinien obrócić się rotor, podejmuje komputer. Wybrany zostanie ten kierunek, przy którym rotor obróci się o mniejszy kąt. Obsługa trybu 1 zawarta jest w liniach 1380—1580.

Tryb 2: Podanie przez użytkownika liczby i kierunku obrotów (linie 1600—1920).

Tryb 3: Podanie przez użytkownika liczby kroków i kierunku obrotów (linie 1940—2190).

Powtarzające się części programu zostały napisane jako podprogramy. Są to:

Wydrukowanie linii nagłówka	linie 2210—2280
Obsługa błędnie wprowadzonych danych	linie 2300—2320
Wysłanie na ekran obliczonych wartości	linie 2340—2410
Wysłanie kolejnych kroków sterowania do portu A	linie 2430—2610

Wszystkie tryby pracy umożliwiają wprowadzenie również liczb niecałkowitych. Komputer oblicza wówczas następny najbliższy, możliwy do przyjęcia kąt. Jeśli w trybie 1 zadamy kąt położenia 20°, a silnik może obrócić się jedynie o wielokrotność 7,5°, to wybrany zostanie do realizacji kąt 22,5°.

Podprogram Transmisja do portu

S = 0 ? — Tak: RETURN

Nie

WY: wygaszenie ekranu, tłusty druk

WY: silnik pracuje

Czy kierunek obrotu w lewo ?

Nie | Tak

Obliczenie następnego kroku do wysłania; S = S − 1

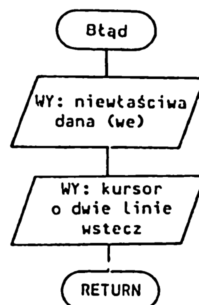
WY: wysłanie następnego kroku do portu A

S = 0 ? — Nie / Tak

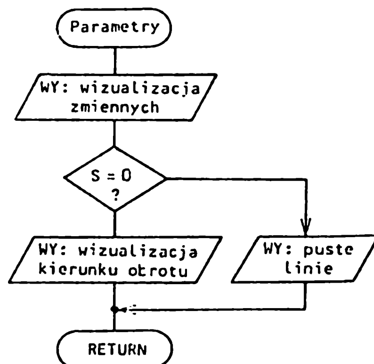
WY: wyłączenie tłustego druku

RETURN

Rys. 3.13. Sieć działań podprogramu wysyłającego słowo sterujące silnikiem do portu A



Rys. 3.14. Sieć działań podprogramu obsługi błędu



Rys. 3.15. Sieć działań podprogramu wyświetlającego parametry sterowania

Rys. 3.16. Program (Basic) "STEROWANIE SILNIKIEM KROKOWYM" (s. 93–97)

```
1000 REM **** PROGRAM : STEROWANIE SILNIKIEM KROKOWYM   ****
1010 REM **** NAZWA PROG. : "STERKROK"                  ****
1020 REM **** NADANIE WARTOSCI ZMIENNYM I ZAPROGRAMOWANIE INTERFEJSU
1030 DIM A(4)
1040 A(1)=9 : A(2)=10 : A(3)=6 : A(4)=5
1050 WI=1 : IW=0 : M=0
1060 OUT 7,128
1070 OUT 4,A(WI)
1080 GOSUB 2200
1090 PRINT "DO DYSPOZYCJI SA TRZY RODZAJE PRACY,"
1100 PRINT "W KTORYCH MOZNA STEROWAC SILNIKIEM"
1110 PRINT "NA ROZNE SPOSOBY."
1120 PRINT "WPROWADZENIE LICZBY UJEMNEJ"
1130 PRINT "POWODUJE POWROT DO 'MENU '"
1140 PRINT
1150 PRINT "PROSZE USTAWIC ROTOR W POLOZENIU ZERO STOPNI"
1160 INPUT "GOTOWE ";AN$
1170 IF LEFT$(AN$,LEN(AN$))<>LEFT$("TAK",LEN(AN$)) THEN  GOTO 1160
1180 REM **** SILNIK KROKOWY - MENU ****
1190 GOSUB 2200
1200 PRINT "WYBOR RODZAJU PRACY :" : PRINT
1210 PRINT "ZADAWANIE KATA OBROTU    (1)"
1220 PRINT "ZADAWANIE LICZBY OBROTOW (2)"
1230 PRINT "ZADAWANIE LICZBY KROKOW  (3)"
1240 PRINT "PRZERWANIE PROGRAMU      (4)"
1250 INPUT "KTORY RODZAJ PRACY ";EG
1260 IF EG<1 OR EG>4 THEN GOSUB 2290 : GOTO 1250
1270 IF EG=4 GOTO 1310
1280 M=EG : S=0 : UM=0 : WD=0 : SO=0
1290 ON M GOSUB 1370,1590,1930
1300 M=0 : GOTO 1190
1310 REM **** OBSLUGA ZAKONCZENIA PROGRAMU ****
1320 OUT 4,0 : GOSUB 2200
1330 PRINT "STEROWANIE SILNIKIEM KROKOWYM ZAKONCZONE"
1340 PRINT "'ZIMNY ' START POPRZEZ 'RUN '"
```

```
1350 PRINT "'GORACY ' START POPRZEZ 'GOTO 1180 '"
1360 END
1370 REM ****  PIERWSZY RODZAJ PRACY  (1)  ****
1380 GOSUB 2200
1390 GOSUB 2330
1400 INPUT "NOWA WARTOSC KATA ";EG
1410 IF EG<0 THEN RETURN
1420 SO=EG
1430 REM **** OBLICZENIE AKTUALNEGO KATA I LICZBY KROKOW ****
1440 SO=ABS(SO)
1450 IF SO>360 THEN SO=SO-360 : GOTO 1450
1460 WD=INT(SO/7.5)*7.5
1470 IF SO-WD>3.75 THEN WD=WD+7.5
1480 IF WD < IW THEN WD=WD+360
1490 WD=WD-IW : DR=1
1500 IF WD>180 THEN DR=-1 :WD=360-WD
1510 IW=IW+WD*DR
1520 IF IW<0 THEN IW=IW+360
1530 IF IW>=360 THEN IW=IW-360
1540 S=WD/7.5
1550 IF DR=1 THEN DR$="PRAWO" : GOTO 1570
1560 DR$="LEWO"
1570 UM=S/48
1580 GOSUB 2430 : GOTO 1380
1590 REM ****  DRUGI RODZAJ PRACY  (2)  ****
1600 GOSUB 2200
1610 GOSUB 2330
1620 INPUT "LICZBA OBROTOW ";EG
1630 IF EG<0 THEN RETURN
1640 UM=EG
1650 UM=EG
1660 INPUT "KIERUNEK OBROTOW W 'PRAWO ' CZY W 'LEWO '";D$
1670 IF LEFT$(D$,LEN(DS$))=LEFT$(PRAWO,LEN(D$)) THEN DR$="PRAWO":DR=1:GOTO 1700
1680 IF LEFT$(D$,LEN(D$))=LEFT$("LEWO",LEN(D$)) THEN DR$="LEWO":DR=-1:GOTO 1700
1690 GOSUB 2290 : GOTO 1660
```

```
1700 REM ****  OBLICZENIE LICZBY KROKOW  ****
1710 GU = INT(UM) : TU = (UM - GU) * 360
1720 S = GU * 48
1730 IF DR = -1 THEN GOTO 1850
1740 REM **** OBLICZENIA DLA 2 RODZAJU PRACY - OBROTY W PRAWO ****
1750 SO = IW + TU
1760 IF SO >= 360 THEN SO = SO - 360
1770 NW = INT(SO/7.5) * 7.5
1780 IF SO - NW = 3.75 THEN NW = NW + 7.5
1790 IF NW < IW THEN  NW = NW + 360
1800 S = S + (NW - IW)/7.5
1810 WD = ABS(NW - IW) : IW = NW
1820 IF IW >= 360 THEN IW = IW - 360
1830 GOSUB 2430 : GOTO 1600
1840 IF NW < IW THEN  NW = NW + 360
1850 REM **** OBLICZENIA DLA 2 RODZAJU PRACY - OBROTY W LEWO ****
1860 SO = IW - TU
1870 IF SO < 0 THEN SO = SO + 360
1880 NW = INT(SO/7.5) * 7.5
1890 IF SO - NW >= 3.75 THEN NW = NW + 7.5
1900 IF IW < NW THEN  IW = IW + 360
1910 S = S + (IW - NW)/7.5
1920 GOTO 1810
1930 REM ****  TRZECI RODZAJ PRACY  (3)  ****
1940 GOSUB 2200
1950 GOSUB 2330
1960 INPUT "PROSZE PODAC LICZBE KROKOW ";EG
1970 IF EG < 0 THEN RETURN
1980 AS = EG
1990 INPUT "KIERUNEK OBROTOW W (PRAWO/LEWO) ";D$
2000 IF LEFT$(D$,LEN(D$))=LEFT$("PRAWO",LEN(D$)) THEN DR$="PRAWO":DR=1:GOTO 2030
2010 IF LEFT$(D$,LEN(D$))=LEFT$("LEWO",LEN(D$)) THEN DR$="LEWO":DR=-1:GOTO 2030
2020 GOSUB 2290 : GOTO 1990
2030 S = INT(AS) : TS = AS - S : UM = 0
2040 IF TS >= 5 THEN S = S + 1
```

```
2050 IF S >= 48 THEN  UM = UM + 1 : S = S - 48 : GOTO 2060
2060 IF DR = -1 THEN GOTO 2140
2070 REM **** OBLICZENIA DLA 3 RODZAJU PRACY - OBROTY W PRAWO ****
2080 SO = IW + (S+TS) * 7.5
2090 IF SO >= 360 THEN SO = SO - 360
2100 WD = S * 7.5 : IW = IW +WD
2110 IF IW >= 360 THEN  IW = IW - 360
2120 S = S + UM * 48 : UM = S/48
2130 GOSUB 2430 : GOTO 1940
2140 REM **** OBLICZENIA DLA 3 RODZAJU PRACY - OBROTY W LEWO ****
2150 SO = IW - ((S + TS) * 7.5)
2160 IF SO < 0 THEN SO = SO + 360
2170 WD = S * 7.5 : IW = IW - WD
2180 IF IW < 0 THEN IW = IW + 360
2190 GOTO 2120
2200 REM **** LINIA NAGLOWKA ****
2210 CLS
2220 PRINT TAB(15);"STEROWANIE SILNIKIEM KROKOWYM"
2230 PRINT TAB(15);"============================="
2240 IF M = 0 THEN PRINT TAB(20);"MENU" : GOTO 2260
2250 PRINT TAB(20);"RODZAJ PRACY";M
2260 PRINT
2270 PRINT
2280 RETURN
2290 REM **** OBSLUGA BLEDOW ****
2300 PRINT "BLEDNA WARTOSC"
2310 PRINT CHR$(27); : PRINT CHR$(27);
2320 RETURN
2330 REM **** WYSWIETLANIE PARAMETROW STEROWANIA SILNIKIEM ****
2340 PRINT "ZADANY KAT          =";SO; TAB(28);"STOPNI"
2350 PRINT "AKTUALNY KAT        =";IW; TAB(28);"STOPNI"
2360 PRINT "OBROTY              =";UM
2370 PRINT "ROZNICA KATOW       =";WD; TAB(28);"STOPNI"
2380 PRINT "LICZBA KROKOW       =";S
2390 IF S = 0 THEN PRINT : RETURN
```

```
2400 PRINT "KIERUNEK OBROTOW     W ";DR$
2410 PRINT : RETURN
2420 REM **** WYSYLANIE DANYCH DO INTERFEJSU ****
2430 IF S = 0 THEN RETURN
2440 CLS : PRINT CHR$(26); CHR$(26); CHRS(26); CHRS(23)
2450 PRINT TAB(5);"SILNIK DZIALA"
2460 PRINT
2470 IF DR = 1 THEN GOTO 2550
2480 REM **** STEROWANIE W LEWO ****
2490 FOR I = 1 TO S
2500 WI = WI + 1
2510 IF WI = 5 THEN WI = 1
2520 OUT 4,A(WI)
2530 NEXT I
2540 RETURN
2550 REM **** STEROWANIE W PRAWO ****
2560 FOR I = 1 TO S
2570 WI = WI - 1
2580 IF WI = 0 THEN WI = 4
2590 OUT 4,A(WI)
2600 NEXT I
2610 RETURN
2620 REM M  RODZAJ PRACY
2630 REM S  LICZBA KROKOW
2640 REM UM LICZBA OBROTOW
2650 REM DR KIERUNEK OBR.
2660 REM WD ROZNICA KATOW
2670 REM SO ZADANY KAT
2680 REM IW AKTUALNA POZ. ROTORA
```

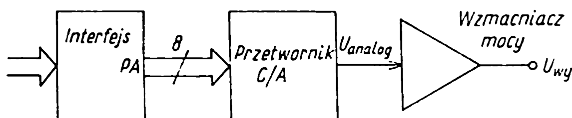
3.3. Programowany zasilacz sieciowy

Parametry

- zakres regulacji napięcia wyjściowego: od 0 do 11 V,
- maksymalny prąd obciążenia: 6 A,
- 256 ustalonych poziomów napięcia,
- żądana wartość napięcia wprowadzona do komputera w postaci cyfrowej.

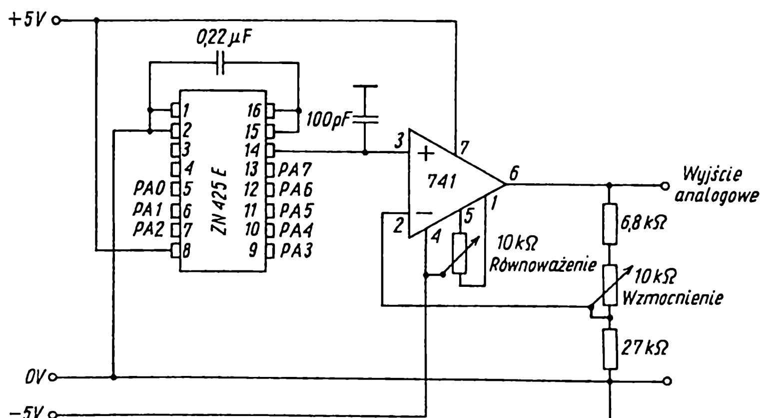
Sprzęt (układ)

Na rysunku 3.17 pokazano schemat blokowy zasilacza. Komputer podaje żądaną wartość napięcia w postaci cyfrowej (liczba binarna) na wejście przetwornika cyfrowo-analogowego, na którego wyjściu pojawia się napięcie odpowiadające tej liczbie. Napięcie to jest doprowadzone do wejścia stopnia mocy z tranzystorem pracującym w układzie wtórnika emiterowego.

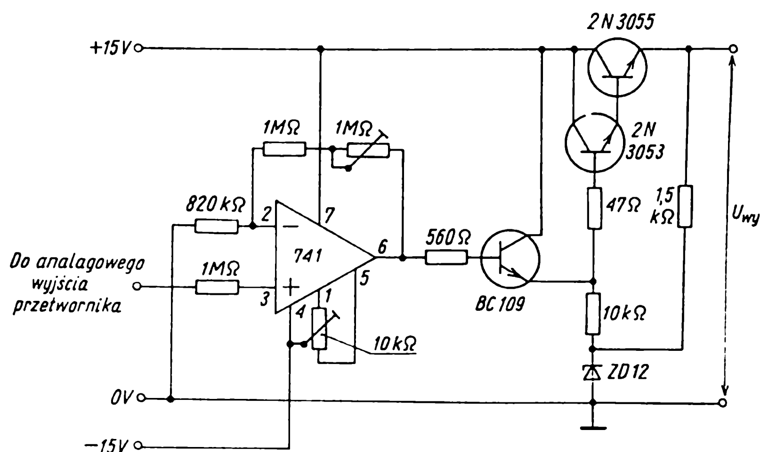


Rys. 3.17. Schemat blokowy programowanego zasilacza

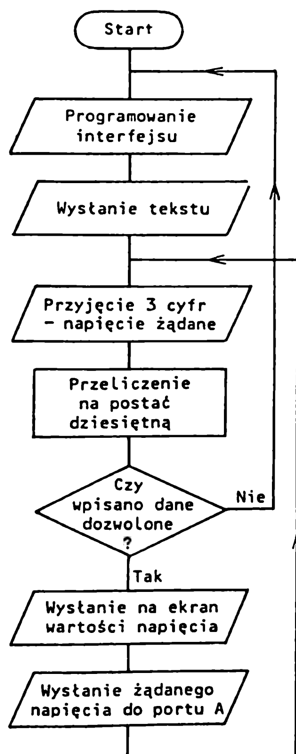
Na rysunku 3.18 przedstawiono układ przetwornika cyfrowo-analogowego, a na rys. 3.19 — stopień mocy zasilacza. Zastosowano tu układ scalony ZN 425 firmy Ferranti, którego działanie zostało omówione w p. 2.11 (schemat blokowy — rys. 2.2). Osiem wejść danych tego przetwornika zostaje wysterowane przez port A interfejsu komputera. Na wyjściu analogowym (wyprowadzenie *14*)



Rys. 3.18. Schemat układu przetwornika cyfrowo-analogowego



Rys. 3.19. Stopień mocy zasilacza programowanego



Rys. 3.20. Zgrubna sieć działań programu sterującego zasilacz

Rys. 3.21. Program (Basic) "ZASILACZ PROGRAMOWANY"

```
100 REM **** PROGRAMOWANY ZASILACZ        ****
110 REM **** NAZWA PROGRAMU : "PROGZAS"   ****
120 CLS : PRINT CHR$(23)
130 OUT 7,128  :  REM PROGRAMOWANIE INTERFEJSU
140 FOR Z = 15680 TO 15744 : POKE Z,32 : NEXT Z
150 PRINT @ 384,"PODAJ WYMAGANE NAPIECIE"
160 PRINT "FORMAT - TRZY CYFRY BEZ PRZECINKA"
170 PRINT @ 522," ";
180 B$ = "" : W = 0
190 PRINT CHR$(14); : FOR N = 1 TO 50 : A$ = INKEY$
200 IF A$ = "" THEN NEXT  N ELSE  GOSUB 240 : NEXT N
210 PRINT CHR$(15); : FOR N = 1 TO 50 : A$ = INKEY$
220 IF A$ = "" THEN NEXT N ELSE GOSUB 240 : NEXT N
230 IF W >= 3 GOTO 290 ELSE GOTO 190
240 IF ASC(A$)=8 AND LEN(B$)>0 THEN B$=LEFT$(B$,LEN(B$)-1):PRINT A$;:RETURN
250 IF ASC(A$) <> 8 THEN B$=B$+A$:PRINT A$;:W=W+1
260 IF W = 2 PRINT @ 528,".";
270 IF W = 3 PRINT @ 534,"WOLTOW"
280 RETURN
290 X$ = LEFT$(B$,3) : B$ = X$
300 C$=LEFT$(B$,1):D$=MID$(B$,2,1):E$=RIGHT$(B$,1)
310 C = ASC(C$) : D = ASC(D$) : E = ASC(E$)
320 C = C - 48 : D = D - 48 : E = E - 48
330 IF C = 0 OR C = 1 GOTO 340 ELSE 120
340 IF C = 1 AND D > 2 GOTO  120
350 IF C = 1 AND D = 2 THEN  E = 0
360 F = C * 10 + D + E / 10 : PRINT
```

```
370 PRINT "NAPIECIE WYJSCIOWE W WOLTACH : ";
380 PRINT F : PRINT @ 522,"             "
390 G = C * 100 + D * 10 + E
400 G = INT( G * 2.1 )
410 OUT 4,G : GOTO 150
420 END
430 REM **** LISTA ZMIENNYCH ****
440 REM ========================
450 REM A$  SPRAWDZANIE KLAWIATURY
460 REM C$  1. ODCZYTANY ZNAK KLAWIATURY(W KODZIE ASCII)
470 REM D$  2. ODCZYTANY ZNAK KLAWIATURY(W KODZIE ASCII)
480 REM E$  3. ODCZYTANY ZNAK KLAWIATURY(W KODZIE ASCII)
490 REM C   DZIESIETNA WARTOSC 1. ODCZYTANEGO ZNAKU
500 REM D   DZIESIETNA WARTOSC 2. ODCZYTANEGO ZNAKU
510 REM E   DZIESIETNA WARTOSC 3. ODCZYTANEGO ZNAKU
520 REM F   WARTOSC NAPIECIA WYJSCIOWEGO
530 REM G   SLOWO BINARNE WYSYLANE DO PORTU
540 REM W   LICZNIK ILE KLAWISZY ZOSTALO PRZYCISNIETYCH
```

ustala się nowy poziom napięcia po czasie konwersji równym ok. 1 μs. Zostaje ono wzmocnione przez pierwszy wzmacniacz operacyjny i podane na wejście stopnia mocy. Ponieważ przetwornik, przy zwiększeniu słowa sterującego o jeden, podwyższa napięcie wyjściowe o 10 mV, maksymalna wartość napięcia wyjściowego wynosi 2,56 V. Konieczne jest więc dodatkowe wzmocnienie, skoro maksymalne napięcie na wyjściu stopnia mocy powinno być równe ok. 12 V. Zastosowana w tym stopniu dioda Zenera współpracująca z tranzystorem BC 109 ogranicza napięcie wyjściowe do ok. 12 V. Z diody tej można zrezygnować, jeśli takie ograniczenie nie jest wymagane. Trzy tranzystory użyte w stopniu mocy powodują przesunięcie poziomu napięcia o ok. 2,1 V. Ponieważ przesunięcie jest praktycznie stałe w całym zakresie, więc napięcia wyjściowe będą o tę wartość za niskie. Błąd ten można jednak łatwo skorygować. W cytowanym rozwiązaniu zastosowano korekcję układową. Błąd jest korygowany przez odpowiednie ustawienie potencjometru kompensacji napięcia niezrównoważenia wzmacniacza operacyjnego.

Program

Na rysunku 3.20 jest pokazana ogólna sieć działań programu, a na rys. 3.21 — program. Działanie: najpierw zostaje wygaszony ekran monitora, ustawione duże znaki graficzne, a następnie zostaje zaprogramowany interfejs (port A jako wyjście!). Pierwszy wysłany tekst jest pytaniem o wartość żądanego napięcia (linie 150—160). Ponieważ wprowadzenie danych odbywa się za pomocą instrukcji INKEY zamiast INPUT, nie muszą być one potwierdzane klawiszem ENTER. Korzyść ta została jednak okupiona bardziej rozbudowanym ciągiem instrukcji programu. Dzieje się tak dlatego, że w trakcie jednej instrukcji INKEY może być odczytana tylko jedna cyfra, w dodatku w kodzie ASCII. Trzy wprowadzone znaki muszą być następnie zdekodowane, a otrzymane cyfry dziesiętne ustawione odpowiednio do ich wagi (linie 190—320). Dodatkowo po drugiej cyfrze powinien pojawić się brakujący przecinek. Po sprawdzeniu, że chodzi o napięcie o dozwolonej wartości (mniejszej niż 13 V; linie 330—350), wartość napięcia zostaje wyświetlona na ekranie (linie 370—380). W liniach 390—410 następuje obliczenie danej wyjściowej dla przetwornika i wysłanie jej do portu A.

Regulacja układu

Po zmontowaniu układu, wprowadzeniu i wystartowaniu programu podajemy jako żądane napięcie 0 V. Za pomocą potencjometru "równoważenie" ustawiamy na wyjściu zasilacza dokładnie 0 V. Następnie podajemy żądaną wartość napięcia 10 V i regulujemy napięcie wyjściowe za pomocą potencjometru "wzmocnienie" oraz rezystora nastawnego 1 MΩ tak, aby uzyskać dokładnie 10 V. Procedurę tę powtarzamy tak długo, aż w obydwu przypadkach napięcia na wyjściu będą prawidłowe. Po tak przeprowadzonej regulacji układ powinien działać poprawnie.

Ulepszenia

Zamiast łączyć obwód sprzężenia zwrotnego w drugim stopniu z wyjściem wzmacniacza operacyjnego, można by go dołączyć do wyjścia mocy (emiter 2N 3055). Podwyższyłoby to liniowość układu oraz zmniejszyło zależność napięcia wyjściowego od prądu obciążenia. Przy niewłaściwym montażu może pojawić się jednak niebezpieczeństwo samowzbudzenia (niekontrolowane oscylacje).

3.4. Sterowanie poprzez zmiany współczynnika wypełnienia (WW)

W przeszłości regulacja obrotów (prędkości obrotowej) silnika prądu stałego była realizowana w ten sposób, że w szereg z silnikiem włączano przełączaną rezystancję. W bardziej luksusowych urządzeniach zastąpiono rezystancję tranzystorem mocy. Umożliwiło to automatyczną regulację lub stabilizację prędkości obrotowej silnika, ale było jednocześnie związane ze zmniejszeniem sprawności, zwłaszcza przy małych obrotach. Ustawiczny wzrost ceny energii elektrycznej powodował, że sprawność urządzeń zyskała na znaczeniu, co z kolei dało pierwszeństwo w zastosowaniu układom o małych stratach.

Takie małostratne sterowanie silnikiem polega na tym, że silnik jest zasilany napięciem impulsowym. Częstotliwość powtarzania impulsów pozostaje przy tym stała, a zmienia się *współczynnik wypełnienia* (stosunek czasu trwania impulsu do okresu powtarzania). Odpowiada to regulacji za pomocą zmiany kąta przepływu (poprzez regulację momentu włączania tyrystora), używanej w tyrystorowych układach zasilających prądu zmiennego. Długi impuls i krótka przerwa dają dużą moc silnika, skracanie impulsu i wydłużanie przerwy powoduje zmniejszenie mocy. Ponieważ tranzystor sterowany impulsowo przyjmuje jedynie dwa stany — zatkania i nasycenia — które charakteryzują się małą mocą strat, w wyniku otrzymujemy wysoką sprawność. Impulsowe zasilanie daje jeszcze jedną korzyść: ponieważ rozruchowy moment obrotowy silnika elektrycznego zależy od przyłożonego napięcia, a w czasie trwania impulsu przyłożone jest pełne napięcie, więc przy każdej prędkości obrotowej otrzymuje się optymalny moment obrotowy.

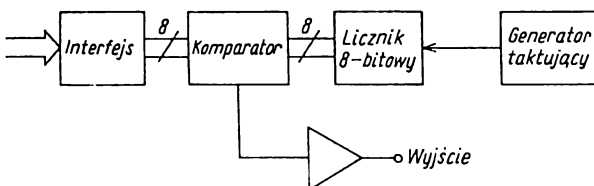
Jeśli chodzi o wybór częstotliwości powtarzania impulsów, to zakres od 100 do 200 Hz okazał się najdogodniejszy. Zbyt małe częstotliwości prowadzą bowiem do dużej nierównomierności obrotów (bicie), natomiast zbyt duże powodują wzrost strat z powodu prądów wirowych w tworniku (ze względu na pętlę histerezy). Częstotliwość przebiegu zasilającego może wywołać także zauważalne efekty akustyczne (pisk, który przenosi się z silnika na przekładnię lub konstrukcję mocującą).

Parametry

- zmiana WW (współczynnika wypełnienia) w 256 krokach,
- zakres zmian WW od 0,4 do 99,6%,
- częstotliwość powtarzania impulsów ok. 200 Hz,
- maksymalny prąd wyjściowy ok. 3 A,
- dwie możliwości do wyboru: WW zmienny lub WW ustalony.

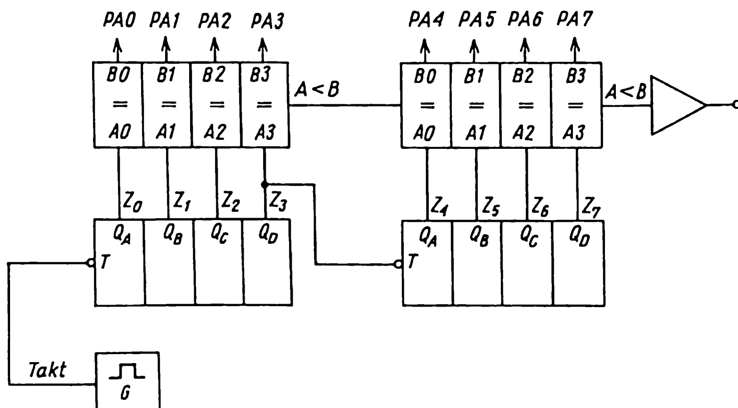
Sprzęt (układ)

Na rysunku 3.22 jest pokazany blokowy schemat sterowania poprzez zmiany współczynnika wypełnienia. Wprowadzona do portu A i zapamiętana w jego buforze wielkość sterująca jest porównywana w komparatorze cyfrowym ze stanem 8-bitowego licznika zliczającego w sposób ciągły impulsy z generatora taktującego (od 0 do 255). Na początku cyklu zliczania na wyjściu układu pojawia się stan wysoki (impuls). Stan taki trwa dopóty, dopóki stan licznika nie osiągnie wartości słowa binarnego zapamiętanego w porcie A. Od tego momentu aż do końca cyklu zliczania na wyjściu układu pozostaje poziom niski (przerwa).

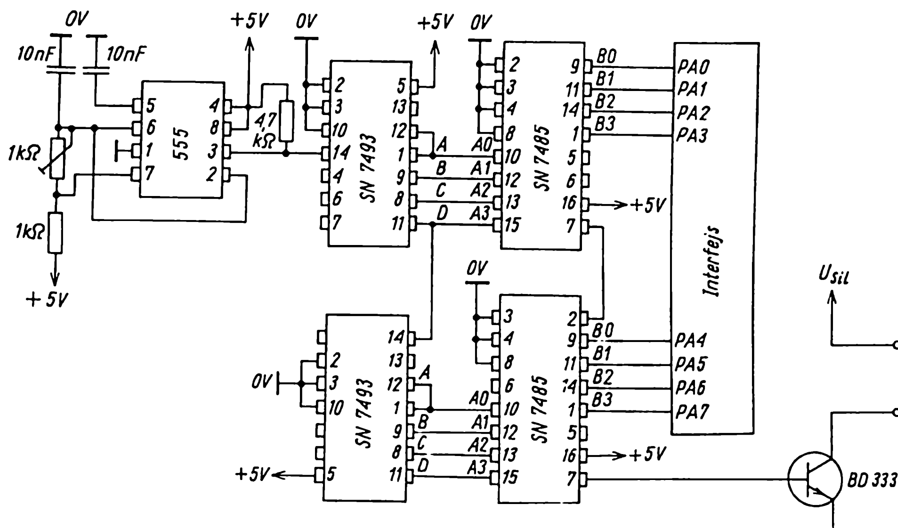


Rys. 3.22. Schemat blokowy sterowania poprzez zmiany współczynnika wypełnienia (WW)

Na rysunku 3.23 pokazano schemat logiczny, a na rys. 3.24 — schemat montażowy układu sterowania WW. Częstotliwość generatora taktującego (układ czasowy 555) jest ustawiana za pomocą rezystora 1 kΩ i wynosi 51,2 kHz.



Rys. 3.23. Schemat logiczny sterowania WW



Rrys. 3.24. Schemat montażowy układu sterowania WW

To daje czas trwania jednego cyklu równy 5 ms, czyli częstotliwość impulsów przebiegu zasilającego równą 200 Hz. Do tej częstotliwości odniesione są także wyprowadzane na ekran informacje o prędkości obrotowej. Napięcie przebicia kolektor-emiter tranzystora mocy Darlingtona zależy od napięcia pracy zastosowanego silnika. Jeśli nie chce się napędzać silnika, to trzeba w obwód kolektora włączyć rezystor.

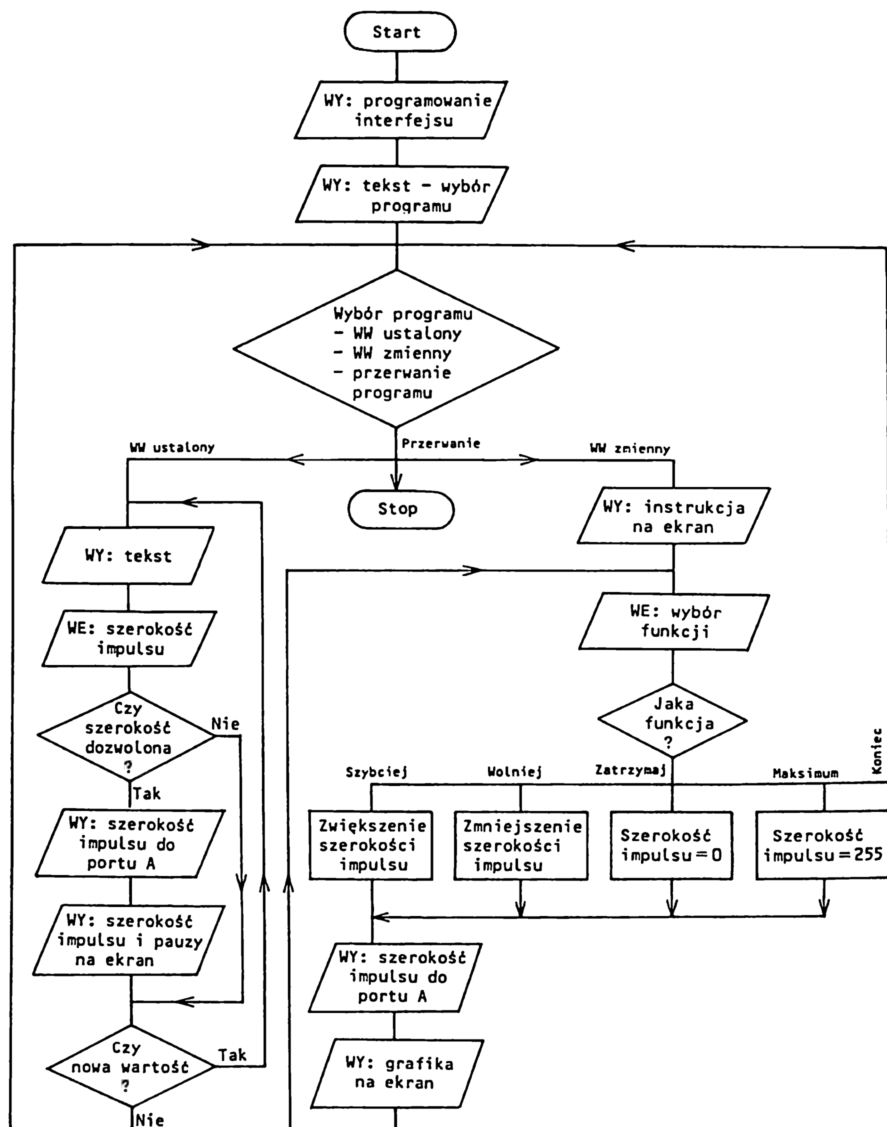
Program

Na rysunku 3.25 przedstawiono zgrubną sieć działań, a na rys. 3.26 — wynikający z niej program w Basicu. Po zaprogramowaniu interfejsu i wyświetleniu "menu" na ekranie (linie 1040—1120) użytkownik może wybrać jeden z trzech wariantów programu:

— wprowadzenie określonego współczynnika wypełnienia,
— zmienny współczynnik wypełnienia,
— przerwanie programu.

W pierwszym przypadku wprowadza się czas trwania impulsu w postaci liczby z zakresu 0—255. Komputer oblicza i wyświetla na ekranie bezwzględną wartość czasu trwania impulsu i przerwy oraz wartość współczynnika wypełnienia w procentach (linie 1270—1330). Jednocześnie do układu zewnętrznego zostaje wysłana odpowiednia wielkość sterująca.

Jeśli zostanie wybrany zmienny współczynnik wypełnienia, można natychmiast — przez przyciśnięcie klawisza *S* (szybciej) lub *W* (wolniej) — zmienić współczynnik wypełnienia o określoną wartość (linie 1640—1760).



Rys. 3.25. Zgrubna sieć działań programu "STEROWANIE WW"

Rys. 3.26. Program (Basic) "STEROWANIE WW" (s. 107–109)

```
1000 REM **** PROGRAM " ZMIANA WSPOLCZYNNIKA WYPELNIENIA"  ****
1010 REM **** NAZWA PROGRAMU : "PULSPAUS"                   ****
1020 OUT 7,128:OUT 4,0
1030 CLS
1040 PRINT @ 10,"WYBOR PROGRAMU"
1050 PRINT @ 74,"--------------"
1060 PRINT @ 202,"PROGRAM 'W '  (USTALONE WYPELNIENIE)"
1070 PRINT @ 330 "PROGRAM 'Z '  (ZMIENNA LICZBA OBROTOW)  "
1080 PRINT @ 394 "----------------------------------------------"
1090 PRINT @ 458,"UWAGA ! PRZYCISNIETY MOZE BYC TYLKO 1 KLAWISZ"
1100 PRINT @ 522,"----------------------------------------------"
1110 PRINT @ 832,
1120 INPUT "WYBOR PROGRAMU (ZAKONCZENIE PROGRAMU - 'K ')",A$
1130 IF A$="W" GOSUB 1190
1140 IF A$="Z" GOSUB 1450
1150 IF A$="K" GOSUB 1180
1160 A$="OZ"
1170 GOTO 1030
1180 END
1190 REM **** PODPROGRAM WSP. WYPEL . ****
1200 CLS
1210 PRINT @ 220," "
1220 INPUT "SZEROKOSC IMPULSU (ZAKRES OD O DO 255)";B
1230 IF B=128 THEN LET B=129
1240 IF B>=256 GOTO 1380
1250 IF B<=-1 GOTO 1380
1260 OUT 4,B
1270 C=INT((5000/255)*B)
1280 Y=INT(5000-C)
1290 CLS
1300 PRINT @ 194,"IMPULS          MIKROSEKUND"
1310 PRINT @ 223,"PRZERWA         MIKROSEKUND"
1320 PRINT. @ 202,C : PRINT @ 232,Y
```

```
1330 PRINT @ 258,"WSP. WYPRL. W % : ";B*100/256;" % ";(256-B)*100/256
1340 PRINT @ 395,"NOWA WARTOSC  ? (T/N)"
1350 INPUT D$
1360 IF D$="T" GOTO 1200
1370 OUT 4,0
1380 CLS
1390 PRINT @ 202,"IMPULS=0000"
1400 PRINT @ 330,"KONIEC PROGRAMU ? (T/N)"
1410 INPUT E$
1420 IF E$="N" GOTO 1200
1430 OUT 4,0
1440 RETURN
1450 REM **** PODPROGRAM STEROWANIA SILNIKIEM  ****
1460 REM ----------------------------------------
1470 Q=7 : O=0 : H=0
1480 CLS
1490 PRINT @ 192,"PREDKOSC OBROTOWA SILNIKA ZMIENIA SIE KLAWISZAMI  S(SZYBCIEJ)"
1500 PRINT @ 269,"                                                   W(WOLNIEJ)"
1510 PRINT@ 333,"                                                   Z(ZATRZYMAJ)"
1520 PRINT @ 397,"                                                   M(MAKSIMUM)"
1530 PRINT @ 522,"                           ZAKONCZENIE PROGRAMU  K(KONIEC)"
1540 FOR X=1 TO 2000 : NEXT X
1550 CLS
1560 H=52
1570 FOR W=6 TO 8
1580 SET(8,W) : SET(112,W)
1590 NEXT W
1600 PRINT @ 259,"MIN"
1610 PRINT @ 311,"MAX"
1620 PRINT @ 896,"S(SZYBCIEJ) W(OLNIEJ) Z(ATRZYMAJ) M(AKSIMUM) K(ONIEC)"
1630 PRINT @ 960,"^           ^           ^           ^           ^"
1640 F$=INKEY$
1650 IF F$="S" GOTO 1710
```

```
1660 IF F$="W" GOTO 1750
1670 IF F$="Z" GOTO 1780
1680 IF F$="M" GOTO 1800
1690 IF F$="K" GOTO 2180
1700 GOTO 1640
1710 H=H+4
1720 IF H=128 THEN LET H=129
1730 IF H>=256 THEN LET H=255
1740 GOTO 1820
1750 H=H-4
1760 IF H<0 THEN H=0
1770 GOTO 1820
1780 H=0
1790 GOTO 1820
1800 H=255
1810 GOTO 1820
1820 OUT 4,H
1830 PRINT @ 323,"   "
1840 PRINT @ 375,"   "
1850 PRINT @ 259,"   "
1860 PRINT @ 311,"   "
1870 IF H=0 GOTO  2020
1880 IF H=255 GOTO 2100
1890 IF H<=0 GOTO 1960
1900 REM
1910 X=INT((H/2)-16)
1920 IF X<=9 GOTO 1950
1930 PRINT @ 468,"
1940 SET(X,Q)
1950 GOTO 2160
1960 X=INT((H/2)-17)
1970 X=X+2
1980 IF X<=9 GOTO 2010
1990 PRINT @ 468,"
2000 RESET(X,Q)
2010 GOTO 2160
2020 S=111
2030 RESET(S,Q)
2040 S=S-1
2050 IF S>=9 GOTO 2030
2060 PRINT @ 323,"!!!"
2070 PRINT @ 468,"SILNIK WYLACZONY"
2080 OUT 4,0
2090 GOTO 2160
2100 I=9
2110 SET(I,Q)
2120 I=I+1
2130 IF I<=110 GOTO 2110
2140 PRINT @ 375,"!!!"
2150 PRINT @ 468,"MAKSYMALNE OBROTY"
2160 O=H
2170 GOTO 1640
2180 OUT 4,0
2190 RETURN
```

Wciśnięcie *H* ustawia minimalną, a wciśnięcie *M* — maksymalną wartość WW (linie 1780—1810). Jednocześnie program przygotowuje grafikę w postaci słupków, dzięki której otrzymuje się łatwą w odczycie wizualizację WW. Jeśli w przedstawionym układzie użyje się silnika prądu stałego, to można przewidzieć dla niego minimum rozruchowe, np.

```
1705 IF H <= 52 THEN H = 52
```

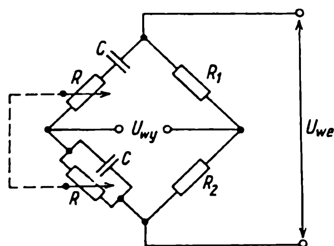
Silniki te wymagają pewnej minimalnej wartości współczynnika wypełnienia w fazie rozruchu.

3.5. Programowany generator częstotliwości

Parametry

- cyfrowo programowany generator sinusoidalny,
- zakres częstotliwości: od 20 do 20460 Hz z kwantem 20 Hz,
- niski współczynnik zawartości harmonicznych,
- wysoka stabilność amplitudy.

Automatyczne pomiary charakterystyk częstotliwościowych filtrów lub wzmacniaczy wymagają stosowania programowanych generatorów częstotliwości. Można je budować używając scalonych generatorów funkcyjnych, np. XR2206. W układzie tym uzyskuje się płynne przestrajanie częstotliwości w wyniku przemiany różnych fal prostokątnych. Częstotliwość reguluje się za pomocą doprowadzonego z zewnątrz napięcia stałego. Zaletą układu jest możliwość uzyskania napięcia wyjściowego o dowolnym kształcie (sinusoidalnego, prostokątnego, trójkątnego lub piłozębnego), wadą natomiast są duże zniekształcenia nieliniowe, zwłaszcza przy większych częstotliwościach. Jeżeli wymagany jest sygnał o bardzo małym współczynniku zawartości harmonicznych (małe zniekształcenia), to zaleca się użycie generatora RC, najlepiej z mostkiem Wiena-Robinsona. Mostek ten jest złożony z dwóch dwójników *RC* (rys. 3.27) i służy jako człon ustalający częstotliwość oscylacji (która zależy jedynie od wartości elementów *R* i *C*). Ponieważ rezystancje łatwo dają się regulować lub



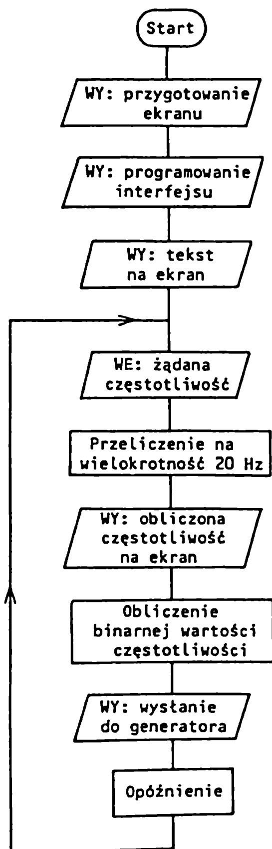
Rys. 3.27. Mostek Wiena-Robinsona

Rys. 3.28. Schemat układu sinusoidalnego generatora *RC*

przełączać, układ ten jest zalecany w przypadku, gdy trzeba cyfrowo sterować zmianami częstotliwości.

Sprzęt (układ)

Na rysunku 3.28 pokazano układ generatora *RC* zaproponowany przez Rittgera [5]. Dwa rezystory *R* z rys. 3.26 zastąpiono dużą liczbą rezystorów, które można łączyć równolegle za pomocą przełącznika scalonego CMOS — CD 4066. W ten sposób uzyskuje się różne wartości rezystancji *R*, a więc i różne częstotliwości drgań generatora. Rezystory te wraz ze wzmacniaczem LF 356 oraz pojemnościami (5,6 nF + 2,7 nF) × 2 tworzą oscylator. Za nim włączono wzmacniacz separujący, który oddziela obciążenie od wyjścia oscylatora. Wadą oscylatorów *RC* są wahania amplitudy napięcia wyjściowego wynikające ze zmian generowanej częstotliwości oraz zmian napięcia zasilającego. Aby usunąć te niedogodności, przewidziano układ stabilizacji amplitudy składający się z trzech wzmacniaczy operacyjnych.



Rys. 3.29. Sieć działań programu "GENERATOR"

Program

Na rysunku 3.29 jest pokazana sieć działań programu sterowania oscylatorem, a na rys. 3.30 — program w języku Basic. Po przygotowaniu ekranu oraz zaprogramowaniu interfejsu (linie 140—210) zostaje wysłane na ekran pytanie o częstotliwość. Po wprowadzeniu, wartość częstotliwości zostaje przeliczona na najbliższą wielokrotność 20 Hz (linie 250 i 260) i wysłana na ekran. Obliczona wielkość sterująca (wielokrotność 20 Hz) jest wysyłana do generatora w postaci

```
100 REM **** PROGRAM "GENERATOR"          ****
110 REM **** NAZWA PROGRAMU:  "GEN"       ****
120 CLS
130 REM LINIE 130-190 : RAMKA NA EKRANIE
140 FOR X = 0 TO 127
150 SET (X,1) : SET (X,47)
160 NEXT X
170 FOR Y = 1 TO 47
180 SET (0,Y) : SET (1,Y) : SET (126,Y) : SET (127,Y)
190 NEXT Y
200 PRINT @ 328,"G E N E R A C J A";
210 OUT 7,136
220 PRINT @ 520,"PODAJ CZESTOTLIWOSC W 'Hz' ";
230 PRINT @ 584,"(OD 20 DO 20460 Hz ) : ";
240 INPUT FS
250 IF FS > 20460           GOTO 220
260 F = INT(FS / 20 + .5)
270 FE = 20 * F
280 PRINT @ 648,"CZESTOTLIWOSC WYNOSI : ";FE;"Hz" ;
290 FO = INT(F / 256)
300 FU = F - FO * 256
310 OUT 4,FU   :  REM WYSLANIE MNIEJ ZNACZACEGO BAJTU
320 OUT 5,FO   :  REM WYSLANIE BARDZIEJ ZNACZACEGO BAJTU
330 FOR  I = 1 TO 500 : NEXT I   :  REM OPOZNIENIE
340 PRINT @ 584,"NOWA CZESTOTLIWOSC W 'Hz' :"
350 GOTO 240
360 REM **** LISTA ZMIENNYCH ****
370 REM ========================
380 REM FS ZADANA CZESTOTLIWOSC
390 REM FE REALIZOWANA CZESTOTLIWOSC
400 REM F  WARTOSC CZESTOTLIWOSCI W POSTACI CYFROWEJ
410 REM FO BARDZIEJ ZNACZACY BAJT STERUJACY
420 REM FU MNIEJ ZNACZACY BAJT STERUJACY
```

Rys. 3.30. Program (Basic) "GENERATOR"

dwóch słów 8-bitowych, których wartości są obliczane w liniach 280 i 290. W słowie bardziej znaczącym (FO) są wykorzystywane jedynie dwa najmłodsze bity, ponieważ sterowanie oscylatora jest realizowane za pomocą słowa 10-bi-

towego. Oba słowa FU i FO zostają przesłane kolejno do portów A i B, a następnie pojawia się na ekranie pytanie, czy użytkownik chciałby wprowadzić nową częstotliwość.

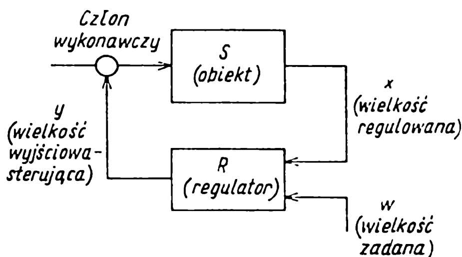
Regulacja układu

Z klawiatury wprowadza się częstotliwość 20 Hz. Następnie zwiera się końcówki źródło-dren tranzystora FET i ustawia potencjometr P_1 w takiej pozycji, żeby układ zaczął generować. Po usunięciu zwory ustala się wartość P_2 tak, aby uzyskać minimalną zawartość harmonicznych w sygnale wyjściowym (maksymalna liniowość charakterystyki tranzystora FET).

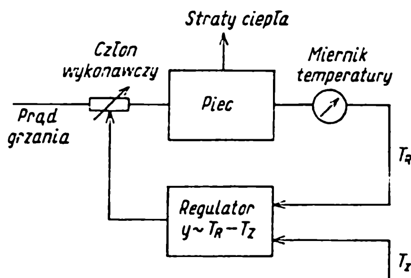
4. Regulacja za pomocą komputera wyposażonego w Basic

4.1. Podstawowe pojęcia techniki regulacji

Technika regulacji jest dziedziną bardzo rozległą. Zrozumienie jej ścisłej teorii, lub też dokładne zaprojektowanie obwodu regulacji ciągłej, wymaga znajomości matematyki wyższej (rachunku różniczkowego, całkowego itd.). Z tego też względu podane tu zostanie jedynie krótkie wprowadzenie. Rysunek 4.1 ilustruje zasadę działania układu regulacji, przy czym oznaczenia są zgodne z zaleceniami normy DIN 19226.



Rys. 4.1. Ogólny schemat blokowy układu regulacji

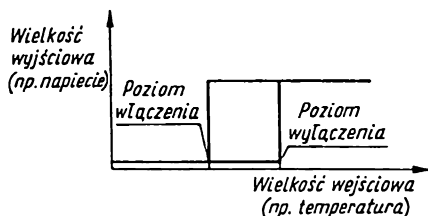


Rys. 4.2. Przykład obwodu regulacji (T_R — temperatura rzeczywista, T_Z — temperatura zadana)

Obiektem regulacji jest przykładowo piec (rys. 4.2), którego temperatura powinna być utrzymywana na stałym poziomie, albo silnik, którego obroty mają być stabilizowane. Regulator otrzymuje sygnał regulowany x, a więc temperaturę albo prędkość obrotową, i porównuje go z sygnałem wartości zadanej w. Z różnicy pomiędzy regulowaną a zadaną wartością sygnałów, czyli z uchybu regulacji, regulator ustala wielkość sygnału sterującego (nastawiającego) y. W

wyżej wspomnianych przypadkach jest to napięcie zasilania silnika albo elementów grzejnych pieca. To, że w ogóle potrzebny jest regulator, wynika z istnienia sygnałów zakłócających. Są to sygnały, które wpływają na sygnał regulowany: w przypadku pieca m.in. straty ciepła np. przez przewodzenie, a w przypadku silnika poprzez zmiany obciążenia lub wahania napięcia zasilającego. Zadaniem regulatora jest jedynie utrzymanie wartości sygnału regulowanego równej wartości zadanej, tzn. uchybu regulacji równego zeru. W rzeczywistości warunek ten może być spełniony jedynie w przybliżeniu.

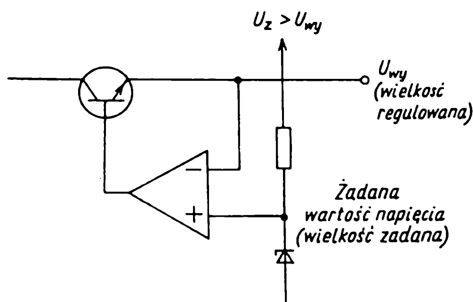
Ze względu na sposób, w jaki regulator reaguje na zmiany sygnału regulowanego, rozróżnia się przede wszystkim układy o działaniu ciągłym i nieciągłym (impulsowym lub przekaźnikowym). Przykładem układu regulacji przekaźnikowej jest elektryczne żelazko do prasowania z termoregulatorem bimetalicznym. Regulatory tego typu są obecnie najbardziej rozpowszechnione. W najprostszej postaci regulator o działaniu nieciągłym przyjmuje tylko dwa stany: włączone lub wyłączone. Stąd też określa się ten typ regulatora jako dwupołożeniowy. Elementy te wykazują różnicę między wielkością sygnału włączającego i wyłączającego, np. włączają przy innej temperaturze niż wyłączają. Ze względu na kształt charakterystyki przejściowej takiego regulatora (rys. 4.3) mówi się o przełączaniu z histerezą. Są także regulatory trójpołożeniowe. Na przykład urządzenie klimatyzacyjne o trzech stanach: ogrzewanie — wyłączone — chłodzenie albo zawór mieszający centralnego ogrzewania, też o trzech stanach: otwieranie (wzrost temperatury) — zatrzymanie stanu chwilowego — zamykanie (obniżanie temperatury).



Rys. 4.3. Charakterystyka statyczna regulatora dwupołożeniowego

O ile regulatory o działaniu nieciągłym mogą przyjmować jedynie stany dyskretne, o tyle w przypadku regulatorów ciągłych wielkość sterująca może przyjmować dowolną wartość wewnątrz przedziału nastaw. Tego typu regulatory są przeważnie stosowane tam, gdzie dopuszczalne są jedynie niewielkie wahania wielkości regulowanej. Ze względu na sposób, w jaki regulator reaguje na zmiany sygnału regulowanego wywołane przez sygnał zakłócający, rozróżnia się trzy podstawowe grupy: regulatory P, I oraz D, które dają się łączyć razem, przy czym jako optymalny otrzymuje się regulator PID.

W przypadku regulatora P (proporcjonalnego) wartość sygnału wyjściowgo (sterującego) jest proporcjonalna do sygnału wejściowego (uchybu). Na tej zasadzie pracują najczęściej stabilizatory napięcia. Rysunek 4.4 ilustruje zasadę działania pętli regulacji. Zachowanie tego regulatora można najłatwiej przedstawić za pomocą charakterystyki skokowej — rys. 4.5 (odpowiedzi na skok jednostkowy). Przy skokowej zmianie wielkości regulującej występuje skokowa, proporcjonalna zmiana sygnału wyjściowego. Regulator typu P jest regulatorem bardzo szybkim. Występujące błędy regulacji nie są całkowicie likwidowane (resztkowy uchyb regulacji).

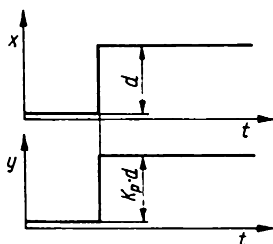


Rys. 4.4. Część regulacyjna stabilizatora napięcia. Napięcie sterujące tranzystor mocy jest proporcjonalne do różnicy między istniejącą a zadaną wartością napięcia wyjściowego

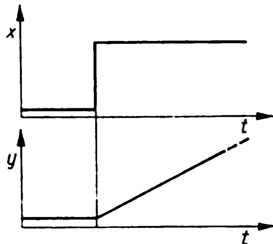
W przypadku regulatora I (całkującego) sygnał sterujący jest proporcjonalny do całki po czasie t z uchybu regulacji. Wynika stąd, że szybkość, z jaką zmienia się wielkość wyjściowa, czyli szybkość reakcji, jest proporcjonalna do sygnału uchybu. Niematematycznie można to wyrazić następująco: skokowa zmiana sygnału regulowanego wywołuje jedynie powolną zmianę sygnału sterującego. Ilustracją jest tu rys. 4.6, na którym przedstawiono charakterystykę skokową regulatora. Regulator I reaguje więc wolniej niż regulator P, ale za to uchyb regulacji dąży do zera. Układ reaguje najczęściej narastaniem lub opadaniem sygnału sterującego, tzn. wielkość regulowana waha się wokół swojej wartości średniej.

W przypadku regulatora D (różniczkującego) wartość sygnału sterującego jest proporcjonalna do szybkości zmiany uchybu regulacji. Jak widać na rys. 4.7 sygnał sterujący w odpowiedzi na skok wielkości regulowanej przyjmuje na chwilę inną wartość, po czym wraca znowu do swojej wartości pierwotnej. Stały uchyb regulacji nie wywołuje żadnego sygnału sterującego, nie może więc zostać zlikwidowany. Z tego powodu regulatory D są stosowane jedynie w połączeniu z innymi regulatorami.

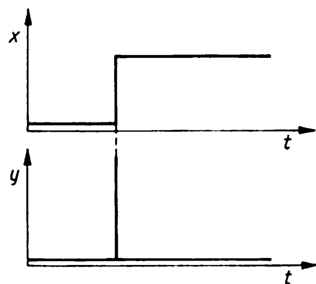
W przeszłości te trzy typy regulacji, albo ich kombinacje, były realizowane prawie wyłącznie przy użyciu układów analogowych, zwłaszcza wzmacniaczy operacyjnych. Obecnie będzie w tej dziedzinie wzrastać zastosowanie komputerów. Ponieważ w takim przypadku wielkość regulowana nie będzie już



Rys. 4.5. Charakterystyka skokowa regulatora P



Rys. 4.6. Charakterystyka skokowa regulatora I



Rys. 4.7. Charakterystyka skokowa regulatora D

pobierana w sposób ciągły, natomiast w dyskretnych odstępach czasu będzie próbkowana, a sygnał sterujący także nie będzie w sposób ciągły korygowany — całki i pochodne w równaniach opisujących sygnał sterujący zamienione zostaną na sumy i ilorazy różnicowe.

Dla tych, którzy orientują się w analogowej technice regulacji, podane zostaną w skrócie równania słuszne dla regulacji cyfrowej (DDC — ang. *Direct Digital Control* — bezpośrednie sterowanie cyfrowe), pozwalające ustalić wartość sygnału sterującego y w i-tym interwale pomiarowym [2]. Laik powinien przynajmniej prześledzić przedstawioną na rys. 4.8 sieć działań programu.

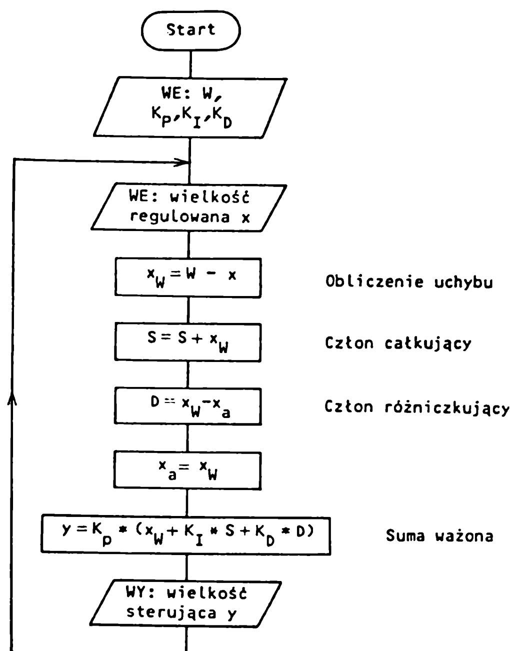
Dla ustalenia wartości sygnału sterującego y, w przypadku dyskretnego próbkowania wielkości regulowanej w odstępach czasu T_a (okres próbkowania), słuszna jest zależność:

$$y_i = K_{PM}\,(X_{Wi} + K_I\,\Sigma X_{Wi} + K_D\,(X_{Wi} - X_{Wi-1}))$$

gdzie: $X_{Wi} = W - X_i$.

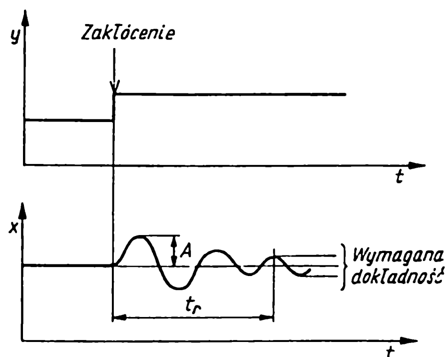
$$K_{PM} = \left(1 + \frac{T_v}{T_n}\right) K_P; \qquad K_I = \frac{T_s}{T_n + T_v}; \qquad K_D = \frac{T_v}{\left(1 + \frac{T_v}{T_n}\right) T_s}$$

Sieć działań programu dla regulatora PID (rys. 4.8) przedstawia obliczenie składników (proporcjonalnego (P), całkowego (I) i różniczkowego (D))sygnału sterującego y, na podstawie uchybu regulacji występującego w kolejnych interwałach pomiarowych (x_a, x_w). W części D można zauważyć brak niektórych ilorazów różnicowych: sygnał uchybu/odstęp między próbkami. Ponieważ okres próbkowania jest stały (T_a), został zatem uwzględniony w stałej K_D. Dobór optymalnych wartości stałych K_P, K_I, K_D musi każdy, kto nie chce wdawać się w teorię, ustalić drogą prób. Co się zaś tyczy optymalizacji, istnieją tu dwie możliwości: optymalna reakcja na zakłócenie albo optymalna reakcja na zmianę

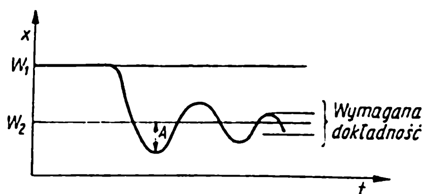


Rys. 4.8. Sieć działań programu "REGULATOR PID"

sygnału wartości zadanej. Optymalna reakcja na zakłócenie oznacza, że sygnał regulowany po przyjściu zakłócenia wykazuje niezbyt duże przeregulowanie, a czas regulacji jest minimalny (rys. 4.9). Oznacza to, że przebieg regulacji jest zakończony w momencie, w którym maksymalna odchyłka wielkości regulowanej nie przekracza wymaganej dokładności pomiaru. W przypadku optymalnej reakcji na zmianę wielkości zadanej, wymaganie jest takie samo, jednak odnosi



Rys. 4.9. Reakcja układu regulacji na zakłócenie (t_r — czas regulacji, A — przeregulowanie)



Rys. 4.10. Odpowiedź układu regulacji na zmianę wielkości zadanej

się do zmiany tej wielkości — tzn. do zmiany wymaganej wartości sygnału regulowanego (rys. 4.10). Jeśli nie jest wymagana realizacja regulatora PID, a jedynie np. regulator PI, to odpowiednią stałą należy uczynić równą zeru, np. $K_D = 0$. Można wówczas pominąć także obliczenie zmiennej D (por. rys. 4.8).

W przypadku obiektu regulacji, który pracuje zasilany napięciem stałym (np. regulacja silników prądu stałego), oddzielny człon wykonawczy sterowany wielkością y może zostać pominięty. Zamiast tego podaje się wstępnie ustaloną składową stałą napięcia albo (co jest znacznie lepsze), o ile znana jest zależność prędkości obrotowej silnika od napięcia zasilającego, oblicza się tę składową stałą dla zadanej prędkości obrotowej, a wartość sygnału sterującego y do niej dodaje. Jeśli realizujemy regulator I (z członem całkującym), to można także sygnał y, po odpowiednim wzmocnieniu mocy, użyć bezpośrednio do sterowania silnika.

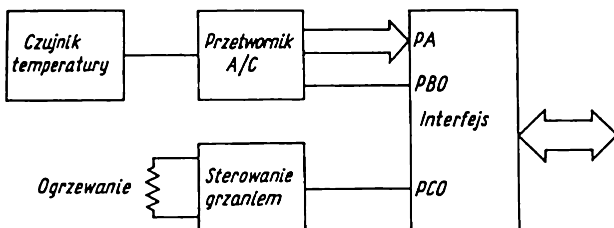
4.2. Regulator temperatury

Parametry

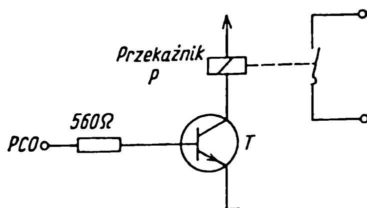
- regulacja dwupołożeniowa,
- temperatura minimalna i maksymalna, wybierane dowolnie,
- zakres temperatur wybierany z przedziału 0—100°C.

Sprzęt (układ regulacji)

Na rysunku 4.11 pokazano schemat blokowy układu regulacji. Część układu, która realizuje próbkowanie wartości chwilowej sygnału regulowanego, odpowiada układowi do pomiaru temperatury, omówionemu w p. 2.1.3 (por. rys. 2.4 i



Rys. 4.11. Schemat blokowy układu regulacji temperatury

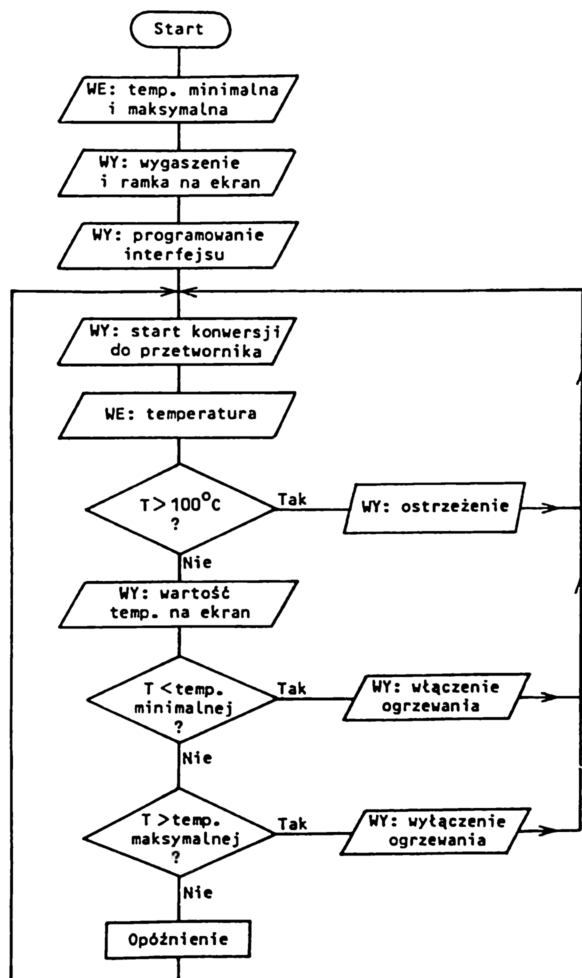


Rys. 4.12. Sterowanie ogrzewaniem. P — przekaźnik włączający ogrzewanie, T — BD 333 lub odpowiednik (tranzystor Darlingtona)

2.12). Rysunek 4.12 przedstawia obwód sterowania grzaniem. Wybór napięcia zasilania tranzystora Darlingtona nie jest krytyczny i zależy od napięcia załączania użytego przekaźnika. Po zmontowaniu urządzenia regulację układu wykonuje się tak jak podano w p. 2.1.3.

Program

Na rysunku 4.13 pokazano sieć działań programu, a na rys. 4.14 — program w języku Basic. Na początku (linie 120—140) ma miejsce wprowadzenie wartości progowych temperatury — minimalnej i maksymalnej (tzw. okno temperaturo-



Rys. 4.13. Sieć działań programu "REGULACJA TEMPERATURY"

```
100 REM **** PROGRAM "REGULACJA TEMPERATURY"          ****
110 REM **** NAZWA PROGRAMU: "REGTEMP"                ****
120 CLS : PRINT "REGULACJA TEMPERATURY" : PRINT
130 INPUT "PODAJ DOLNA GRANICE TEMPERATURY "; MI
140  INPUT "PODAJ GORNA GRANICE TEMPERATURY "; MA
150 CLS
160 REM LINIE 170-220 RAMKA NA EKRANIE
170 FOR X = 0 TO 127
180 SET (X,1) : SET (X,47)
190 NEXT X
200 FOR Y = 1 TO 47
210 SET (0,Y) : SET (1,Y) : SET (126,Y) : SET (127,Y)
220 NEXT Y
230 PRINT @ 264," **** REGULACJA TEMPERATURY **** ";
240 PRINT @ 392,"MINIMALNA TEMPERATURA";MI;"STOPNI"
250 PRINT @ 456,"MAKSYMALNA TEMPERATURA";MA;"STOPNI"
260 OUT 7,144   :   REM PROGRAMOWANIE INTERFEJSU
270 OUT 5,0 : OUT 5,1
280 T = INP(4)
290 IF T>100 PRINT@ 648,"PRZEKROCZENIE ZAKRESU";:GOTO 270
300 PRINT @ 648,"TEMPERATURA WYNOSI    ";
310 M$="###"
320 PRINT @ 712,USING M$;T;
330 PRINT " STOPNI CELSJUSZA"; : PRINT @ 776,"OGRZEWANIE";
340 IF T < MI THEN OUT 6,1 : PRINT @ 787," WLACZONE ! ";
350 IF T > MA THEN OUT 6,0 : PRINT @ 787," WYLACZONE !";
360 FOR I = 1 TO 100 : NEXT I : REM OPOZNIENIE
370 GOTO 270
```

Rys. 4.14. Program (Basic) "REGULACJA TEMPERATURY"

we). Dalej następuje przygotowanie ekranu (wygaszanie, zapalenie ramki itd., linie 170—250). Teraz rozpoczyna się cykl pracy układu regulacji: po wysłaniu rozkazu "start pomiaru" oraz odczycie zmierzonej wartości temperatury (linie 270—330), porównuje się ją z wartością zadaną (linie 340 i 350). Zależnie od wyniku porównania, ogrzewanie zostanie włączone lub wyłączone.

4.3. Regulacja prędkości obrotowej

Zadaniem tego układu regulacji jest utrzymanie na stałym poziomie (stabilizacja) prędkości obrotowej silnika, niezależnie od zmian obciążenia. Chodzi tu o układ regulacji ciągłej.

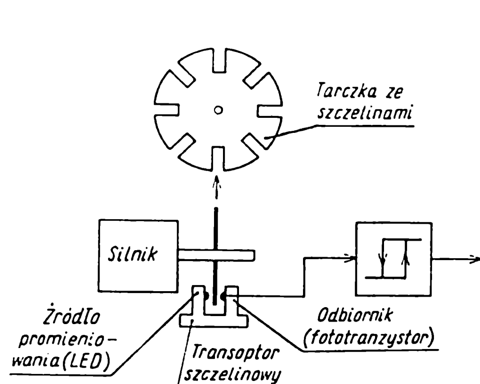
Parametry

- regulacja w zakresie od 500 obr/min do 6000 obr/min,
- wymagana prędkość obrotowa jest wprowadzana z klawiatury i można ją zmieniać w dowolnym momencie,
- zatrzymanie silnika (z klawiatury) w dowolnym momencie,

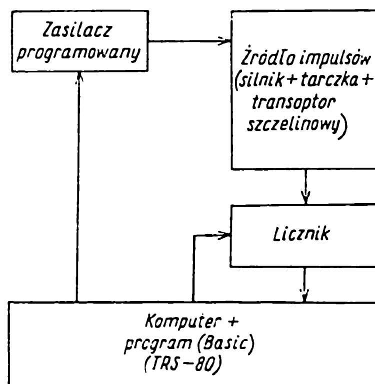
- sygnalizacja przekroczenia zadanej maksymalnej dopuszczalnej prędkości obrotowej,
- wydruk protokółu regulacji.

Sprzęt (układ regulacji)

Na osi silnika, którego prędkość obrotowa ma być regulowana, jest zamocowana okrągła tarczka. Brzeg tej tarczki z ponacinanymi szczelinami lub karbami, przesuwa się w szczelinie transoptora (rys. 4.15). Impulsy dostarczone z transoptora podczas obrotów silnika są zliczane przez licznik w określonym czasie — zwanym czasem bramkowania. Zarówno czas zliczania, jak też impulsy zerujące licznik zapewnia komputer TRS-80. Przed wyzerowaniem zostaje odczytany oczywiście stan licznika, po czym następuje wizualizacja zmierzonej liczby obrotów i — jeśli trzeba — skorygowane napięcie zasilania silnika.



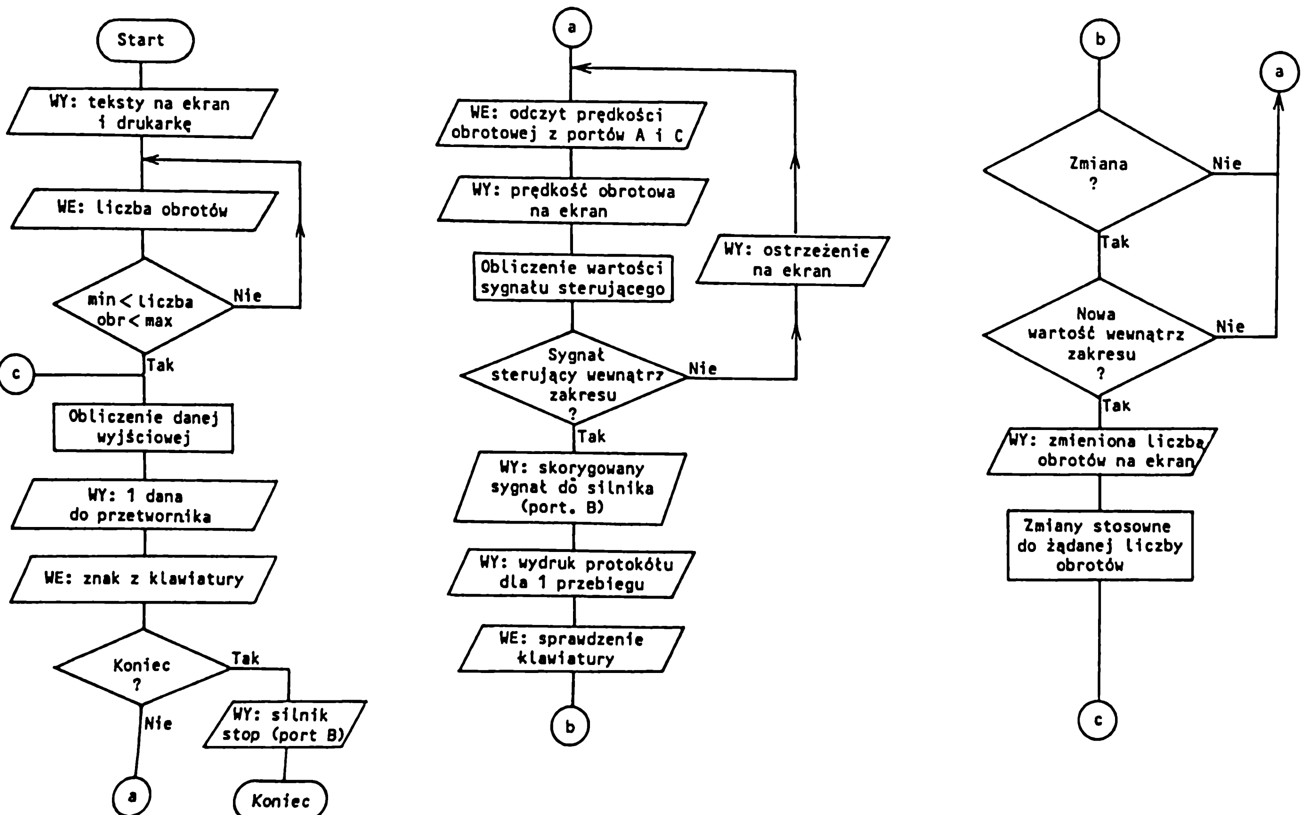
Rys. 4.15. Pomiar prędkości obrotowej za pomocą tarczki ze szczelinami



Rys. 4.16. Schemat blokowy układu regulacji prędkości obrotowej

Na rysunku 4.16 jest pokazany schemat blokowy, a na rys. 4.17 — pełny schemat ideowy układu. Część usytuowana po prawej stronie schematu jest już znana z rozdziału "Pomiar prędkości obrotowej" (p. 2.1.6). W wymienionym rozdziale można znaleźć informacje dotyczące jej elementów oraz ich funkcji. Blok sterowania silnikiem zawiera scalony przetwornik cyfrowo-analogowy ZN 425E. Przetwarza on cyfrowo zakodowaną wartość napięcia sterującego, dostarczoną przez komputer, na proporcjonalne napięcie stałe, które zostaje wzmocnione przez pierwszy wzmacniacz operacyjny 741. Wzmocnienie to jest niezbędne, ponieważ maksymalne napięcie wyjściowe przetwornika wynosi zaledwie 2,5 V, co nie wystarcza do uzyskania maksymalnych obrotów silnika. Drugi wzmacniacz (741) ma jednostkowe wzmocnienie napięciowe i służy do niskoomowego (napięciowego) sterowania tranzystora mocy Darlingtona BD 333. Takie dopasowanie jest szczególnie polecane, ponieważ iskrzenie komuta-

Rys. 4.17. Kompletny schemat ideowy układu



Rys. 4.18. Ogólna sieć działań programu "REGULACJA OBROTÓW"

Rys. 4.19. Program (Basic) "REGULACJA OBROTÓW" (s. 126–129)

```
1000 REM **** PROGRAM " REGULACJA OBROTOW "          ****
1010 REM NAZWA PROGRAMU : "REGOBR"                   ****
1020 REM *********************************************
1030 CLS
1040 COUN = 0 : A = 0 : B$ = STR$(A) : NUM = 839 : NDREH = 0
1050 OUT 7,145
1060 GOSUB 1830
1070 LPRINT "************************************"
1080 LPRINT "**** REGULACJA OBROTOW SILNIKA ****"
1090 LPRINT "************************************"
1100 LPRINT ""
1110 LPRINT ""
1120 LPRINT "***********************************************************
1130 LPRINT "25.01.83*";TAB(10)"CZAS * ";TAB(20)"WART.NAP * ";TAB(36)"ODCHYLKA *
1140 LPRINT "***********************************************************
1150 PRINT @ 900,"ABY ZAKONCZYC NZLEZY PRZYCISNAC DOWOLNY ZNAK OPROCZ CYFRY !!";
1170 PRINT @ 546,"                              ";
1180 PRINT @ 392,"LICZBA OBROTOW SILNIKA MOZE BYC W "
1190 PRINT @ 456,"ZAKRESIE OD 500 DO 6000 OBR/MIN ! "
1200 PRINT @ 520,"PROSZE PODAC LICZBE OBROTOW ";:INPUT AE
1210 DREHZ=AE
1220 FOR X = 27 TO 29 : SET(0,X) : SET(1,X) : SET(126,X) : SET(127,X) : NEXT X :
1230 IF DREHZ < 500 GOTO 1170 :      REM WPROWADZENIE NOWEJ DANEJ
1240 IF DREHZ > 6000 GOTO 1170 :     REM WPROWADZENIE NOWEJ DANEJ
1250 NDREH = DREHZ/23.52941176# :    REM OBLICZENIE PROPORCJ. WART. DLA PRZETWOR.
1260 NDREH = NDREH + .5 :            REM ZAOKRAGLENIE
1270 NDREH = INT(NDREH) :            REM PRZETWORZENIE NA LICZBE CALKOWITA
1280 IF NDREH < 50 THEN NDREH = 50 :REM KOREKCJA ZE WZGLEDU NA MIN OBROTY
1290 OUT 5,NDREH
1300 E$ = INKEY$ :                   REM SRAWDZANIE CZY 'KONIEC
1310 IF E$ = "" GOTO 1330
1320 IF E$ = "!" GOTO 1790
1330 OUT 7,10 : OUT 7,11
```

```
1340 OUT 7,13 :                     REM ZEZWOLENIE DLA 'AND '
1350 FOR B = 1 TO 135 : NEXT B :    REM ZLICZANIE 500 MILISEK
1360 OUT 7,12 :                     REM ZAKAZ DLA 'AND '
1370 ZAEL = 0 : OZAEL = 0 :
1380 ZAEL = INP(4) :                REM ODCZYT 8 MNIEJ ZNACZACYCH BITOW LICZNIKA
1390 OZAEL = INP(6) :               REM ODCZYT 4 BARDZIEJ ZNACZACYCH BITOW
1400 OZAEL=OZAEL AND 15 :           REM WYDZIELENIE 4 MNIEJ ZN. BITOW PORTU C
1410 OZAEL = OZAEL*256 :            REM KOREKCJA WARTOSCI
1420 WERT = 0
1430 WERT = ((ZAEL+OZAEL)/16)*120 : REM OBLICZENIE LICZBY OBROTOW NA MINUTE
1440 IF WERT <= 10 GOSUB 1900 :     REM ZEROWANIE JESLI SILNIK MIEDZY SZCZEL.
1450 PRINT @ 648,"LICZBA OBROTOW NA MINUTE";
1460 M$ = "#####.####"
1470 PRINT @ 670 USING M$;WERT ;
1480 ABWEI = (DREHZ-WERT)*.5 :      REM OBLICZENIE UCHYBU
1490 REGEL = ABWEI/23.52941176#     REM OBLICZENIE WARTOSCI KOREKTY
1500 NDREH = REGEL + NDREH :        REM KOREKCJA WIELKOSCI WYJSCIOWEJ
1510 NDREH = NDREH + .5 :           REM ZAOKRAGLENIE
1520 NDREH = INT(NDREH) :           REM PRZETWORZENIE NA LICZBE CALKOWITA
1530 IF NDREH > 255 GOTO 1730 :     REM PRZEKROCZENIE ZAKRESU DANYCH PRZETW.
1540 IF NDREH < 0 GOTO 1760 :       REM PRZEPELNIENIE UJEMNE
1550 IF NDREH < 25 THEN NDREH = 25: REM KOREKCJA DANEJ WYJSCIOWEJ
1560 OUT 5,NDREH                    REM PRZESLANIE DO PRZETWORNIKA
1570 SPAN = NDREH*(5.65/232)+.45 :  REM OBL. WART. NAP. PRZETWORNIKA
1580 ABWEI = ABWEI * (-2) :         REM KOREKCJA UCHYBU
1590 LPRINT ND;TAB(9) RIGHT$(TIME$,8);" *";TAB(20)SPAN;TAB(32)"V *";ABWEI;TAB(43
1600 A$=INKEY$:IF A$=""THEN GOTO 1300:REM JESLI NIE WCISNIETY KLAWISZ - NOWY ODC
1610 PRINT @ 839,"          ";      REM WYGASZENIE MIEJSCA NA WSKAZANIA
1620 B$ = B$ + A$ : A = VAL(B$) :   REM ZLOZENIE CIAGU ZNAKOW I KONWERSJA
1630 PRINT @ NUM,A; :               REM WSKAZANIE ZMIANY
1640 COUN = COUN + 1 :              REM ZLICZANIE WPROWADZANYCH ZNAKOW
1650 IF A = 0 THEN GOTO 1790 :      REM SKOK DO ZAKONCZENIA PROGRAMU
1660 IF COUN >= 4 THEN GOTO 1670 ELSE 1300 : REM PO 4 ZNAKACH PRZYJECIE DANEJ
```

```
1670 EINGA = 0: EINGA = VAL(B$) : NUM = 839 : REM PRZYJECIE DANEJ
1680 IF EINGA > 6000 THEN GOTO 1720 : REM DANA ZA DUZA - ZEROWANIE I NOWY ODCZYT
1700 MA$ = "####.##"
1710 DREHZ = EINGA:PRINT @ 548,USING MA$;DREHZ ;:COUN = 0 : A = 0 : B$ = STR$(A)
1720 A=0 : B$=STR$(A) : COUN =0 : EINGA+0 : GOTO 1300 : REM ZEROWANIE
1730 PRINT @ 776,"UWAGA PRZEPELNIENIE PRZETW. !!! "; : NDREH = 200 : OUT 5,NDREH
1740 FOR AD = 1 TO 200 : NEXT AD :PRINT @ 776,"                                        ";
1750 GOTO 1300
1760 PRINT @ 776,"UWAGA NIEDOPELNIENIE PRZETW. !!!"; : NDREH = 26 : OUT 5,NDREH
1770 FOR AC = 1 TO 200 : NEXT AC :PRINT @ 776,"                                        ";
1780 GOTO 1300
1790 NDREH = 0 : OUT 5,NDREH
1800 CLS : PRINT @ 447,CHR$(23)"PONOWNY START INSTRUKCJA 'RUN '"
1810 END
1820 REM RAMKA NA EKRAN
1830 FOR X = 0 TO 127
1840 SET(X,1) : NEXT X
1850 FOR Y = 1 TO 47
1860 SET(0,Y) : SET (1,Y) : SET(126,Y) : SET (127,Y) : NEXT Y
1870 FOR X1 = 0 TO 127
1880 SET(X1,47) : NEXT X1
1890 RETURN
1900 WERT = 0
1910 RETURN
1920 REM
1930 REM L I S T A  Z M I E N N Y C H
1940 REM ============================
1950 REM  COUN     LICZNIK SPRAWDZEN INKEY$
1960 REM  NUM      POZYCJA NA EKRANIE
1970 REM  NDREH    AKTUALNA DANA WYJSCIOWA DLA PRZETWORNIKA
1980 REM  DREHZ    ZADANA LICZBA OBROTOW NA MINUTE
1990 REM  ZAEL     ODCZYTANE 8 MLODSZYCH BITOW LICZNIKA
2000 REM  OZAEL    ODCZYTANE 4 STARSZE BITY LICZNIKA
```

```
2010 REM  WERT    AKTUALNA LICZBA OBROTOW
2020 REM  ABWEI   ROZNICA POMIEDZY ZADANYMI A AKTUALNYMI OBROTAMI
2030 REM  REGEL   SKORYGOWANA WARTOSC DO POPRZEDNIEGO PRZESLANIA
2040 REM  SPAN    NAPIECIE PRZESLANE DO ZASILANIA SILNIKA
2050 REM  EINGA   PRZYJETA DANA PRZY ZMIANIE OBROTOW POPRZEZ INKEY$
2060 REM  M$      MASKA DLA ZADANEJ LICZBY OBROTOW
2070 REM  MA$     MASKA DLA ZMIENIONEGO WSKAZANIA LICZBY OBROTOW
2080 REM  A$      WARTOSC INKEY$ DLA ZMIANY LICZBY OBROTOW
2090 REM  B$      SUMA CIAGU ZNAKOW WPROWADZONYCH PRZEZ INKEY$
2100 REM  A       WARTOSC WPROWADZONEGO CIAGU ZNAKOW
2110 REM  E$      WARTOSC INKEY$ SPRAWDZANIA CZY KONIEC ?
2120 REM  AE      WPROWADZONA NA POCZATKU POPRZEZ 'INPUT ' LICZBA OBROTOW
```

tora silnika wywołuje znaczne zakłócenia, które dostawszy się na przetwornik C/A mogą powodować wadliwą pracę całego układu. Część tych zakłóceń daje się wprawdzie zlikwidować przez zastosowanie kondensatora włączonego równolegle z silnikiem, ale nie można w ten sposób zlikwidować składowej sumacyjnej sygnału zakłócającego (patrz rozdz. 7).

Proste silniki prądu stałego wymagają pewnej minimalnej wartości momentu obrotowego do rozruchu, a więc minimalnej wartości napięcia sterującego. W tym przypadku nieco poniżej 1,5 V otrzymuje się minimalną liczbę obrotów ok. 500 obr/min! Dla maksymalnej prędkości obrotowej tzn. 6000 obr/min oraz 256 poziomów dyskretnych napięcia wyjściowego z przetwornika, otrzymujemy minimalny skok liczby obrotów o 40 obr/min.

Program

Działanie programu jest następujące. Po wytworzeniu czasu bramkowania, odczytaniu stanu licznika oraz ustaleniu różnicy między rzeczywistą a wymaganą prędkością (sygnał uchybu), odpowiednio do wielkości tej różnicy skorygowane zostanie napięcie wysyłane do silnika. Teraz program może już wyzerować liczniki, aby mógł rozpocząć się następny pomiar. Na rysunku 4.18 jest pokazana zgrubna sieć działań, a na rys. 4.19 listing tego programu (w języku Basic).

Po starcie programu na drukarkę wysyłany jest nagłówek drukowanego protokółu (linie 1070—1140). Na ekranie pojawia się ramka oraz informacja, jak duże obroty można wprowadzić (linie 1150—1200). Można teraz podać żądaną prędkość obrotową. Po upewnieniu się, czy wprowadzona dana należy do przedziału dozwolonych prędkości, program oblicza wymaganą wartość napięcia silnika, którą wysyła na wejście przetwornika (linie 1230—1290).

Teraz następuje sprawdzenie klawiatury, czy zgłoszono żądanie zatrzymania silnika. Jeśli tak, to silnik zostanie zatrzymany (linie 1300—1320). W przeciwnym przypadku następuje pomiar prędkości obrotowej silnika (czas bramkowania — linie 1330—1360). Zawartość licznika obrotów jest odczytywana w postaci dwóch bajtów z portu A oraz C, przeliczana i wysyłana na ekran w postaci prędkości obrotowej (linie 1380—1470). Potem obliczane jest nowe napięcie zasilania silnika i wysyłane do przetwornika (linie 1480—1550). Jeśli zmierzona liczba obrotów na minutę przekracza przewidziane maksimum, to na ekran wysyłane jest ostrzeżenie.

Regulacja układu

Potencjometr dostrojczy 10 kΩ, którym reguluje się wzmocnienie pierwszego wzmacniacza operacyjnego trzeba ustawić tak, aby można było uzyskać maksymalną prędkość 6000 obr/min. Tę liczbę obrotów wprowadza się z klawiatury po starcie programu i sprawdza, czy silnik osiąga przewidywaną liczbę obrotów.

5. Automatyczne pomiary za pomocą komputera wyposażonego w Basic

5.1. Charakterystyka diody

Parametry

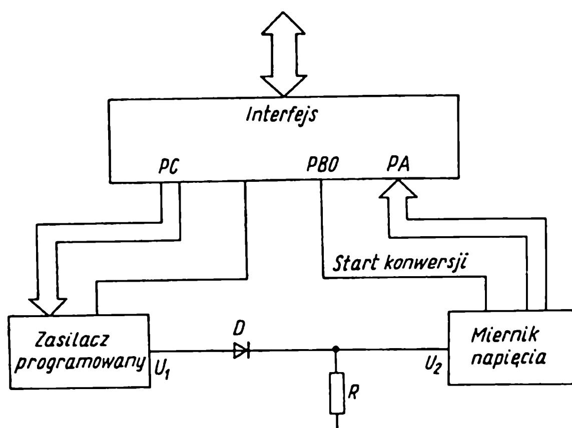
- zakres napięć dla zdejmowanych charakterystyk od 0 do 11 V,
- możliwość pomiaru charakterystyk innych dwójników lub czwórników.

Sprzęt (układ pomiarowy)

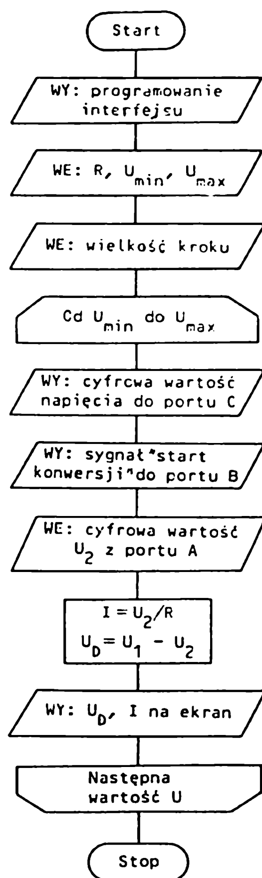
Na rysunku 5.1 pokazano schemat blokowy zestawu pomiarowego. Zestaw ten składa się w zasadzie z zasilacza programowanego oraz miernika napięcia, który powinien mierzyć wartość prądu płynącego przez diodę. Pomiar ten jest realizowany jako pomiar spadku napięcia na szeregowej rezystancji *R*. Wprawdzie wartość tej rezystancji nie jest ograniczona, ponieważ pomiar charakterystyki uwzględnia występujący w niej spadek napięcia, jednak użycie zbyt dużej rezystancji *R* prowadzi do zawężenia zakresu zmian prądu diody. Jako programowane źródło napięcia może być zastosowany układ z rys. 3.18 i 3.19 (patrz p. 3.3), a jako miernik napięcia — układ z rys. 2.4 i 2.7 (patrz p. 2.1.1), jednak należy zwrócić uwagę na zmiany w wyprowadzeniach portów (rys. 5.1). Ponieważ do dyspozycji są tutaj jedynie trzy porty, zrezygnowano z przewidzianego na rys. 2.7 sprawdzania statusu przetwornika.

Program

Rysunek 5.2 przedstawia sieć działań programu "ZDEJMOWANIE CHARAKTERYSTYKI", a rys. 5.3 — program w języku Basic. Po zaprogramowaniu interfejsu trzeba wprowadzić wartość rezystancji *R*, minimalną i maksymalną wartość napięcia zdejmowanej charakterystyki oraz odstęp punktów pomiarowych (linie 140—170). Potem cały zadeklarowany zakres pomiarowy zostanie obsłużony w pętli programowej krok po kroku, przy czym w jednym obiegu pętli



Rys. 5.1. Schemat blokowy zestawu pomiarowego



Rys. 5.2. Sieć działań programu

następuje kolejno: wysłanie wymaganej wartości napięcia do zasilacza w postaci cyfrowej (użyty w linii 190 współczynnik 21 jest narzucony przez wartość wzmocnienia wzmiacniacza zasilacza programowanego), pomiar i odczyt spadku napięcia na rezystorze *R* (linie 180–280), a następnie obliczenie napięcia na diodzie przez odjęcie spadku napięcia na *R*.

Ulepszenia

Opisany powyżej system został wprawdzie przewidziany do pomiaru charakterystyki diody, ale przy niewielkich zmianach w układzie pomiarowym i programie można go użyć np. do pomiaru charakterystyki tranzystora.

Przy odpowiednich zmianach lub uzupełnieniach w programie, będzie również możliwe graficzne przedstawienie na ekranie zdjętej charakterystyki.


```
100 REM **** PROGRAM "POMIAR CHARAKTERYSTYK" ****
110 REM **** NAZWA PROGRAMU : "CHAR"         ****
120 OUT 7,144   REM  PROGRAMOWANIE INTERFEJSU
130 CLS
140 INPUT "REZYSTANCJA R W 'OMACH' :";R
150 PRINT "U-MIN - U-MAX "
160 INPUT "ZAKRES OD 0 DO 12 WOLTOW "; MI , MA
170 INPUT "WIELKOSC KROKU POMIAROWEGO W WOLTACH "; SW
180 FOR U1 = MI TO MA STEP SW
190 U0 = INT( U1 * 21 )
200 OUT 6,U0 : REM ZADANIE WARTOSCI NAPIECIA
210 OUT 5,0  : REM ROZPOCZECIE KONWERSJI
220 OUT 5,1
230 U2 = INP (4) * .01
240 I = U2 / R
250 UD = U1 - U2
260 PRINT "UD =";UD;"WOLTOW","I =";I;"AMPEROW"
270 FOR N= 1 TO 100 : NEXT N : REM OPOZNIENIE
280 NEXT U1
290 REM **** LISTA ZMIENNYCH ****
300 REM ========================
310 REM MI  MINIMALNA WARTOSC NAPIECIA ZASILACZA
320 REM MA  MAKSYMALNA WARTOSC NAPIECIA ZASILACZA
330 REM SW  ODSTEP PUNKTOW POMIAROWYCH
340 REM U1  AKTUALNE NAPIECIE WYJSCIOWE ZASILACZA
350 REM U0  CYFROWA WARTOSC NAPIECIA ZASILACZA
360 REM I   PRAD PLYNACY PRZEZ MIERZONY ELEMENT
370 REM UD  NAPIECIE NA ELEMENCIE MIERZONYM
```

Rys. 5.3. Program (Basic) "POMIAR CHARAKTERYSTYK"

6. Zastępowanie sprzętu oprogramowaniem

W zasadzie sprzęt (ang. *hardware*) można zastępować programem (ang. *software*) i odwrotnie: np. bramki AND lub OR przez użycie instrukcji AND lub OR; przerzutniki bistabilne przez użycie znaczników (ang. *flags*), które mogą być ustawione lub zerowane; rejestry przez zmienne (języka Basic), których wartość może być zmieniana (np. inkrementacja — zliczanie); wreszcie czas trwania impulsu przerzutnika monostabilnego przez pętlę opóźnienia. Rozpatrzmy teraz zalety i wady rozwiązań sprzętowych i programowych.

6.1. Zalety i wady rozwiązań programowych

Zalety

— Oszczędność elementów

Można zrezygnować z układów scalonych wraz z niezbędną płytką drukowaną i podstawkami.

— Możliwość użycia elementów nieliniowych

Często czujniki pomiarowe o liniowej charakterystyce są znacznie droższe niż elementy nieliniowe. Wyniki pomiarów można linearyzować programowo, dzięki czemu celowe jest stosowanie czujników nieliniowych.

— Większa elastyczność

Zmiana funkcji urządzenia jest osiągalna dużo szybciej poprzez zmiany lub uzupełnienia instrukcji programu niż poprzez wymianę układów scalonych lub zmianę połączeń, która w przypadku płytek drukowanych jest prawie niemożliwa.

Wady

— Zwiększone zapotrzebowanie na pamięć

Ze względu na postępujący w ostatnich latach spadek cen scalonych pamięci półprzewodnikowych, nie jest to wada poważna.

— Wydłużenie czasu pisania programu oraz testowania

Ponieważ z zasady zastępujące sprzęt części programu — ze względu na szybkość działania — muszą być napisane w języku wewnętrznym maszyny, trzeba poświęcić im więcej czasu, tym bardziej że z powodu większego prawdopodobieństwa popełnienia błędów, trzeba je bezwarunkowo przetestować.

— Mniejsza szybkość przetwarzania

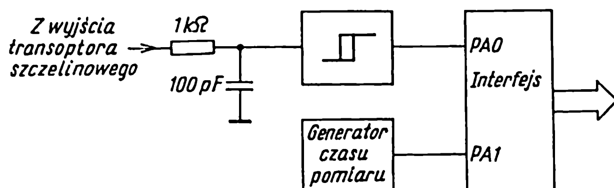
Ta wada decyduje o przyjęciu bądź odrzuceniu rozwiązania programowego. Jeżeli użyjemy instrukcji w języku Basic, to czasy wykonania pojedynczych rozkazów leżą w zakresie milisekundowym. Użycie instrukcji języka wewnętrznego (maszynowego) zmniejszy je do zakresu dziesiątek mikrosekund, podczas gdy rozwiązania sprzętowe zapewniają czas operacji ok. 100 ns.

Z porównania wymienionych wyżej zalet i wad wynika, że rozwiązania programowe należy stosować wszędzie tam, gdzie umożliwiają one uzyskanie wymaganej szybkości przetwarzania.

6.2. Przykłady zastosowań

Oszczędność elementów

W rozdziale 2 pokazano, jak można zrealizować pomiar szybko zmieniającej się wielkości, przechodząc do programowania w języku wewnętrznym. W związku z tym omówiony w p. 2.2.5 pomiar częstotliwości daje się wprost zastosować w przypadku pomiaru (p. 2.1.6 — rys. 2.25) lub regulacji (p. 4.3 — rys. 4.17) prędkości obrotowej. Oszczędność elementów polega na tym, że zamiast pomiaru częstotliwości za pomocą zewnętrznego licznika, liczba szczelin tarczki przesuwających się przez transoptor jest zliczana przez komputer. Na rysunku 6.1 pokazano jedno z możliwych rozwiązań sprzętowych. W obydwu wyżej



Rys. 6.1. Uproszczone rozwiązanie sprzętowe układu do pomiaru lub regulacji prędkości obrotowej

wspomnianych przypadkach rozwiązanie to pozwala zaoszczędzić trzy scalone liczniki. Potrzebny tu dodatkowo generator czasu bramkowania może być złożony z bramek NAND albo zbudowany z jednej pozostałej bramki Schmitta układu SN 7414. Jeśli w grę wchodzi szczególnie wysoka stabilność szerokości impulsu, można zastosować dodatkowy element — układ 555[1].

Linearyzacja

Nieliniowe przetworniki są na ogół najtańsze. Jeśli mimo użycia takiego przetwornika chce się uzyskać dostateczną dokładność, to trzeba obarczone błędem wartości zmierzone skorygować programowo. Jako przykład niech posłuży tutaj czujnik temperatury w postaci rezystora NTC. Choć nie jest to może najlepszy przykład, otrzymujemy tu niedrogi półprzewodnikowy czujnik temperatury. W pewnych zastosowaniach przeszkadza jednak wydzielająca się w nim moc zasilania.

Aby można było wprowadzić korekcję odczytanych wartości zmierzonej temperatury, najpierw musi być sporządzona tablica, która będzie zawierać zależność między temperaturą zmierzoną elektronicznie a rzeczywistą. Zależność tę sporządza się jednorazowo, po zmontowaniu systemu pomiarowego, przy użyciu zwykłego termometru. Poniższy przykład wyjaśnia metodę postępowania.

Temperatura zmierzona elektronicznie	Rrzeczywista temperatura (zmierzona termometrem)
0	1
1	2
2	4
3	5
⋮	⋮
10	11
⋮	⋮
20	19

[1] Polski odpowiednik: ULY7855N (przyp. tłum.).

Z tej tablicy otrzymuje się w Basicu jednowymiarową tablicę:

```
T(0)  = 1
T(1)  = 2
T(2)  = 4
T(3)  = 5
       .
       .
       .
T(10) = 11
       .
       .
       .
T(20) = 19
```

Jeśli zostanie użyty standardowy interfejs, a temperatura będzie odczytywana z portu A, to proponuje się następujący prosty prógram w Basicu (numery linii można sobie uzupełnić dowolnie):

```
I = INP(4)
TEMP = T(I)
```

Jeżeli np. zostanie odczytana, jako temperatura, liczba 3, to komputer znajdzie na czwartym miejscu tablicy (czyli pod T(3)) właściwą temperaturę — tutaj 5°C.

Do większości dostępnych w handlu czujników producenci dołączają karty katalogowe, w których jest zamieszczona charakterystyka przetwarzania. Mając taką charakterystykę lub sporządzając ją samodzielnie na podstawie szeregu pomiarów, łatwo jest wprowadzić programową korekcję odczytanych danych pomiarowych.

Transkoder

Jeśli komputer i urządzenie peryferyjne (np. drukarka) używają różnych kodów, trzeba przy wymianie lub przesyłaniu danych przetwarzać jedne kody na drugie (np. ASCII na EBCDIC). Zwykle stosuje się do tego celu dekoder, najczęściej w postaci pamięci ROM. Można taką konwersję zrealizować również programowo. Ponieważ kody 7-bitowe zawierają nie więcej niż 128 słów, a 8-bitowe — 256, więc niezbędne staje się stworzenie jednowymiarowej tablicy o tej właśnie długości. W języku Basic można każdy ze znaków kodu źródłowego potraktować jako indeks elementu tablicy. W ten sposób, tak zaadresowane miejsce w pamięci zawiera znak kodu wynikowego.

Przykład

kody	
EBCDI	ASCII
F0	30
F1	31
F2	32
.	.
.	.
.	.

Stąd w Basicu tablica (jednowymiarowa) ma następującą postać:

E(F0) = 30
E(F1) = 31
E(F2) = 32
. .
. .
. .

Takie rozwiązanie jest polecane jedynie wówczas, gdy nie jest wymagana duża szybkość przetwarzania. W przeciwnym przypadku należy przejść do poziomu języka maszynowego. Wtedy element kodu wynikowego znajduje się w komórce pamięci, której adres powstaje przez dodanie heksadecymalnej wartości elementu kodu źródłowego do adresu początkowego tablicy kodowej (adresowanie względne). Jeśli przykładowo lista kodów zaczyna się od adresu 8000_{16}, to znak ASCII, którego odpowiednik w kodzie EBCDI ma wartość $F0_{16}$, będzie umieszczony pod adresem $80F0_{16}$.

Układ czasowy (ang. *timer*)

Jeżeli zadanie jest postawione w ten sposób, że komputer musi czekać bezczynnie przez określony czas, zanim będzie mógł odebrać lub wysłać jakieś sygnały i czas ten leży w przedziale od 10 ms do kilku sekund, to wówczas zamiast układu czasowego czy przerzutnika monostabilnego zaleca się zastosować rozwiązanie programowe. Tego rzędu czasy opóźnienia można zrealizować w języku Basic za pomocą pętli FOR..NEXT; np.

```
100 FOR I = 1 TO 1000 : NEXT I
```

W zależności od komputera lub zastosowanego systemu operacyjnego, ta pętla opóźnienia może zapewnić czas oczekiwania od 400 ms do 2,7 s (ostatnia wartość dotyczy TRS-80). Czas opóźnienia można zmieniać dowolnie przez zmianę liczby obiegów pętli. Najmniejsza możliwa do uzyskania zmiana to czas jednego obiegu pętli, który — zależnie od komputera i systemu operacyjnego — zawiera się w granicach od 0,4 do 2,7 ms, co przy opóźnieniu równym 1 s stanowi 0,27% zmiany.

Jeśli ta dokładność nie wystarcza, to można do pętli włączyć "pustą" operację, np. nadanie wartości zmiennej: A = 100 lub operację arytmetyczną, jak np. A = A + 1. Czas obiegu pętli wydłuży się, a więc będzie potrzebna mniejsza liczba obiegów dla tego samego czasu opóźnienia, można za to — zależnie od okoliczności — zapewnić pożądany czas z większą dokładnością. Z drugiej strony biorąc, puste operacje pozwalają uzyskać znacznie większe czasy opóźnienia. Jeśli ktoś pracuje z dyskowym systemem operacyjnym (DOS), może w przypadku dłuższych czasów sprawdzać programowy zegar komputera. Bliższe informacje zaczerpnąć można z p. 3.1, zwłaszcza poleca się przejrzeć program z rys. 3.3.

Ze względu na różnice między komputerami, minimalny czas opóźnienia trzeba ustalić doświadczalnie: zadaje się dużą liczbę obiegów, np. 10 000, i ze zmierzonego stoperem czasu całkowitego oblicza się czas jednego obiegu pętli. Jeśli jednakże podczas oczekiwania realizowanego przez pętlę opóźnienia mają być wykonywane przez komputer inne, określone operacje, to nie jest wówczas możliwe uzyskanie dokładnego interwału czasowego, chyba że w pętli znajduje się część programu bez rozgałęzień, a więc o stałym i dającym się obliczyć czasie wykonania.

7. Testowanie

Kiedy układ jest już całkowicie zmontowany, a właściwy program wprowadzony do komputera, mamy zwykle nadzieję, że urządzenie "ruszy od pierwszego włączenia". Niestety, najczęściej nadzieje się nie spełniają. Przyczyny mogą być różne. Ścieżki obwodów drukowanych mają niekiedy mikropęknięcia, które trudno jest wykryć i zlokalizować, często zdarzają się zimne lutowania. Gorzej, jeśli popełniono błąd w projekcie! W przypadku połączeń wykonanych przewodem mogą wystąpić błędne połączenia (np. do innej nóżki układu scalonego) lub niedogrzane miejsca lutowań (tzn. izolacja przewodu nie odparowała). (Połączenia w takich układach wykonuje się w ten sposób, że końcówki układu scalonego lub podstawki owija się cienkim przewodem i następnie oblutowuje).

Aby nie tracić czasu na chaotyczne sprawdzanie po omacku (dla układu scalonego o 24 końcówkach daje to 2^{24} możliwości), powinno się testować urządzenie systematycznie, albo — aby mieć pewność, że uruchomienie zakończy się sukcesem — zaleca się, zwłaszcza początkującemu, rozłożyć układ na moduły funkcjonalne i każdy z nich uruchamiać oddzielnie. Jeśli wszystkie moduły pracują poprawnie, następuje połączenie ich ze sobą, a test kompletnego urządzenia jest już zwykle mało kłopotliwy. Układ na ogół działa jak należy.

Rozpatrzmy teraz, jako konkretny przykład, układ do pomiaru temperatury. Najpierw podział na moduły — początkujący konstruktor powinien przede wszystkim wybierać małe moduły, np. układ scalony wielkiej skali

Rys. 7.1. Układ do pomiaru temperatury — podział na moduły funkcjonalne

integracji lub dwa do trzech standardowych układów TTL albo MOS. Na rysunku 7.1 pokazano moduły w postaci schematu blokowego.

Nasuwają się tu dwie możliwości: montaż oraz test modułów w kolejności: *A-B-C-D*, albo odwrotnie. Wybieramy wariant pierwszy, bowiem w wariancie drugim (*D-C-B-A*) musielibyśmy najpierw testować program komputerowy. Oznacza to, że trzeba by zasymulować odpowiednie stany logiczne na ośmiu liniach danych oraz na liniach sterowania — np. za pomocą przełączników, co zwiększyłoby koszty. Wybrana zatem pierwsza wersja przedstawia się następująco:

Najpierw montuje się czujnik temperatury i za pomocą dołączonego do jego wyjścia woltomierza (najlepiej cyfrowego) odpowiednio reguluje łącznie z ustawieniem napięcia przesunięcia charakterystyk. Teraz można już przystąpić do montażu i uruchomienia przetwornika. Najpierw trzeba dołączyć układ scalony ZN 425, a na jego odpowiednie wejście (*4*) podać sygnał taktujący. Na wyjściu analogowym (*14*) powinno pojawić się widoczne na oscyloskopie napięcie piłokształtne, przy czym przy szybkiej podstawie czasu można zauważyć, że zbocze narastające składa się z 256 schodków. Jeśli nie mamy do dyspozycji oscyloskopu, należy zapewnić odpowiednio małą częstotliwość taktowania, a do wyjścia *14* dołączyć woltomierz. Jeśli ta próba zakończy się pomyślnie, możemy uznać, że moduł *B* jest w porządku.

Teraz przystępujemy do uruchomienia modułu *C*, czyli układu sterowania przetwornikiem. Zmontowany układ przyłączamy do przetwornika i podajemy ręcznie rozkaz startu konwersji (poziom niski — L). Jeśli na wejście "+" wzmacniacza operacyjnego zostanie przyłożone napięcie np. równe 1 V, to napięcie schodkowe (mierzone na wyprowadzeniu *14*) powinno zatrzymać się na 1 V. Jeśli nie — oznacza to, że jest niewłaściwie ustawiony potencjometr regulacji napięcia niezrównoważenia. Na wyjściu statusu powinien pojawić się wówczas stan niski.

Jako ostatni moduł (programowy moduł *D*) zostanie wprowadzony program i będą usunięte błędy programu — głównie błędy syntaktyczne, które w trakcie wprowadzania programu zgłasza interpreter Basicu. Właściwie po dołączeniu do komputera poprzednio zmontowanej części pomiarowej (moduły *A-C*) całe urządzenie powinno od razu zacząć pracować poprawnie. Błąd mógłby się pojawić jedynie w przypadku uszkodzenia interfejsu (co łatwo daje się sprawdzić za pomocą prostego programu testującego) lub jeśli np. podczas wprowadzania programu zostały pominięte niektóre linie.

8. Zakłócenia

8.1. Wprowadzenie

W czasach, gdy do sterowania stosowało się wyłącznie przekaźniki, problem zakłóceń w dzisiejszym sensie nie istniał. Jeśli mówiło się o zakłóceniu, to zwykle chodziło o przerwy w pracy urządzenia. Sygnały zewnętrzne wprawdzie istniały, ale nie wywoływały zakłóceń, ponieważ czas ich trwania był zbyt krótki, a energia przez nie dostarczana zbyt mała, aby spowodować błędne włączenia przekaźników. Wraz z wprowadzeniem elektroniki (tranzystory i tyrystory, jako elementy przełączające, oraz cyfrowe układy logiczne itd.) pojawiły się nagle znaczne problemy zakłóceniowe, co sprawiło, że wielu projektantów urządzeń elektrycznych zaryzykowało przejście na układy elektroniczne jedynie z dużą rezerwą. To właśnie rozwój elementów wymagających małej mocy zasilania, a więc o dużych poziomach rezystancji z jednej strony i wzrost szybkości działania, a więc wzrost częstotliwości granicznych tych elementów z drugiej strony, doprowadził do tego, że nawet bardzo wąskie impulsy szpilkowe o małej energii powodować mogą poważne zakłócenia w systemach cyfrowych. Tak więc, jeśli tylko układ próbny z luźno poprowadzonymi połączeniami zastąpi się układem z porządnie związaną wiązką kablową albo rezystancję symulującą silnik w układzie sterowania zastąpi silnikiem, to okazuje się, że urządzenie przestaje działać.

Nie pomaga, niestety, klasyczna metoda usuwania zakłóceń przez stosowanie kondensatorów blokujących lub prowadzenie przewodów w ekranie. Wręcz przeciwnie, skutek jest zwykle odwrotny. Częstokroć zakłócenia stają się jeszcze bardziej uporczywe, ponieważ indukcyjności przewodów zasilających tworzą z dołączonymi pojemnościami obwody rezonansowe, których rezonans może przypadkowo wypaść w pasmie sygnałów zakłócających.

Aby oszczędzić czytelnikowi niemiłych doświadczeń zostaną pokrótce opisane przyczyny powstawania zakłóceń oraz podstawowe metody ich usuwania. Ponieważ najczęściej źródła występujących zakłóceń nie można wyodrębnić i

nie ma na to żadnej uniwersalnej metody, jest się zwykle zdanym na własne doświadczenie.

8.2. Rozchodzenie się sygnałów zakłócających

Każde zagadnienie związane z zakłóceniami można rozłożyć na trzy składniki (rys. 8.1):

— Źródła lub przyczyny powstawania zakłóceń: silniki (iskrzenie komutatora), wyłączniki urządzeń elektrycznych, przekaźniki, transformatory, nadajniki radiowe, wyładowania atmosferyczne, uderzenia prądu, impulsy napięciowe na przewodach doprowadzających napięcie zasilające.



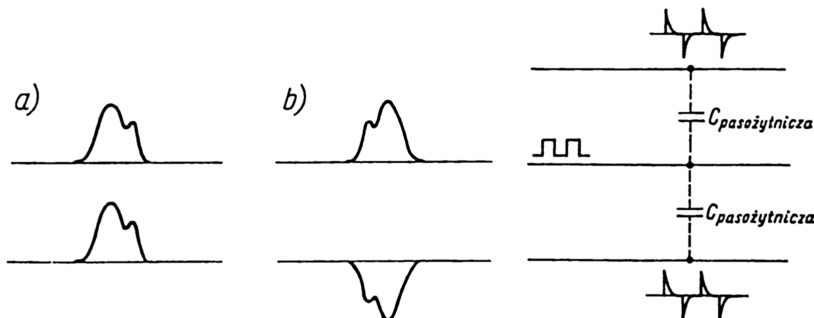
Rys. 8.1. Rozchodzenie się sygnałów zakłócających

— Sposób przenoszenia się zakłóceń: pola elektryczne (sprzężenia pojemnościowe), pola magnetyczne (sprzężenia indukcyjne), pola elektromagnetyczne (promieniowanie wysokiej częstotliwości), przewody (sprzężenia galwaniczne).

— Odbiorniki zakłóceń: wzmacniacze, odbiorniki TV i radiowe, komputery, urządzenia cyfrowe regulacji i sterowania itd.

8.3. Rodzaje zakłóceń i przyczyny ich powstawania

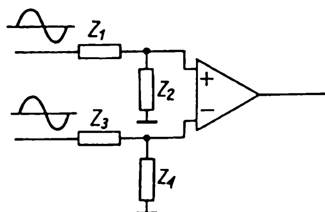
Rozróżnia się dwie składowe sygnałów zakłócających: *współbieżną i różnicową* (rys. 8.2). Sygnały współbieżne (nazywane też *sumacyjnymi*) powstają wówczas, gdy np. przewód będący źródłem zakłócenia przebiega równolegle do dwóch



Rys. 8.2. Sygnały zakłócające: a) współbieżne, b) różnicowe

Rys. 8.3. Powstawanie zakłóceń współbieżnych

przewodów wejściowych, a zakłócenia są przenoszone poprzez sprzężenie pojemnościowe, tzn. wskutek istnienia pojemności pasożytniczych między przewodami (rys. 8.3). Są to najczęściej występujące sygnały zakłócające, przy czym częstotliwości tych sygnałów leżą zwykle w zakresie od 1 do 20 MHz. Sygnały sumacyjne są groźne głównie dlatego, że skutkiem niesymetrii układu powodują powstawanie składowej różnicowej (rys. 8.4). Sygnały zakłócające różnicowe są sygnałami o mniejszych częstotliwościach, zwykle w zakresie od 0,1 do 2 MHz.

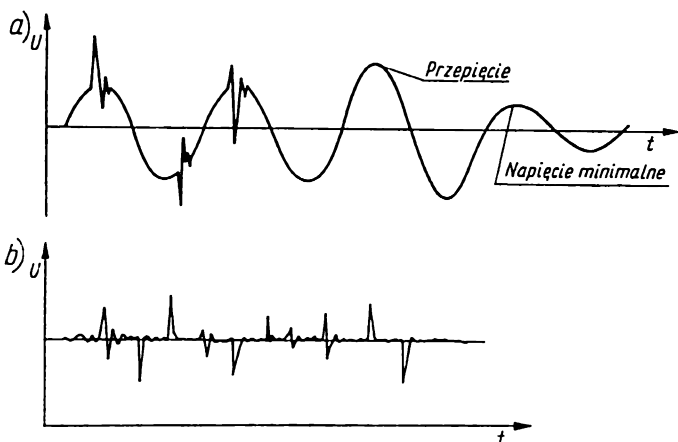


Rys. 8.4. Powstawanie sygnału różnicowego z sygnału współbieżnego, wskutek niesymetrii układu wejściowego (Z_1:Z_2 ≠ ≠ Z_3:Z_4) (np. różne rezystancje wejściowe)

Zakłócenia powstają:

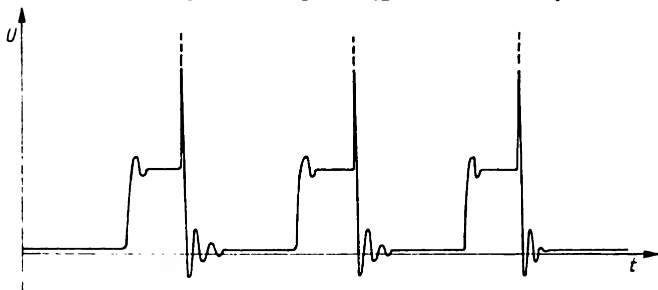
— w przewodach zasilających sieciowych, podczas szybkich przebiegów włączania bądź wyłączania — zwłaszcza przy dużych obciążeniach (rys. 8.5a). Przy wyłączaniu prądu płynącego przez obciążenie o charakterze indukcyjnym, mogą powstawać kilowoltowe impulsy napięcia o pasmie od ok. 0,1 do 3 MHz;

— podczas iskrzenia kolektorów silników (rys. 8.5) — powstaje wówczas sygnał zakłócający o bardzo szerokim pasmie;



Rys. 8.5. Sygnały zakłócające: a) na doprowadzeniach sieciowych, b) na doprowadzeniach zasilających prądu stałego

— podczas załączania i wyłączania obciążeń indukcyjnych przełączanych impulsowo. Nowoczesne elementy półprzewodnikowe charakteryzują się bardzo krótkimi czasami przełączania, z tego względu powstające przy wyłączaniu impulsy napięcia mogą osiągać wartości rzędu kilowoltów (rys. 8.6);



Rys. 8.6. Sygnały zakłócające występujące przy impulsowym przełączaniu obciążeń indukcyjnych

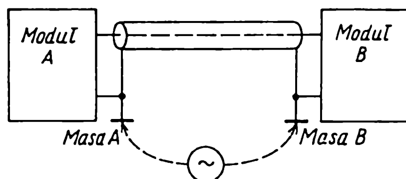
— skutkiem przepływu prądów impulsowych w przewodach doprowadzających napięcie zasilające stałe lub sieciowe. Jeśli stosuje się płytki drukowane, to należy się liczyć z indukcyjnością ścieżki przewodzącej wynoszącą ok. 100 nH na 10 cm długości. Na podstawie zależności

$$U = -L\frac{\Delta I}{\Delta t}$$

otrzymuje się dla zmiany prądu o 100 mA w czasie 10 ns impuls napięcia o amplitudzie 2 V, licząc doprowadzenie tam i z powrotem. W przypadku elementu dużej mocy pobierającego prąd 1 A skok napięcia wyniesie 20 V. Załóżmy teraz, że układ jest zasilany przez skrętkę, o długości 50 cm, wykonaną z dwóch odcinków linki o średnicy 1 mm. Licząc, że taki podwójny przewód ma indukcyjność ok. 600 nH/m, otrzymamy dla wyżej przyjętych wartości napięcie indukowane równe 6 V, przy czym polaryzacja impulsu może być ujemna lub dodatnia, zależnie od tego, czy powstaje on wskutek włączania czy wyłączania prądu. Jeśli nie zapewni się odpowiedniego zabezpieczenia, to zastosowanie elementów o dużym prądzie rozruchu, jak silniki lub żarówki wskaźnikowe, może prowadzić do straty danych, np. w przerzutnikach rejestrów;

— wskutek rozładowań elektrostatycznych. Poruszanie się po podłożu ze sztucznego tworzywa powoduje gromadzenie się ładunków elektrycznych na ciele człowieka (wzrost potencjału do ok. 20 kV). Przy dotknięciu klawiatury komputera lub jego obudowy następuje szybkie rozładowanie w postaci iskry. Natężenie prądu wzrasta wówczas z prędkością 4 A/ns. Można łatwo obliczyć, jak duże impulsy napięcia powstaną na zwykłym kablu połączeniowym między komputerem a klawiaturą;

— na skutek różnicy potencjałów mas (pętla uziemieniowa). Gdy dwie połączone ze sobą galwanicznie części układu są dodatkowo niezależnie od siebie uziemione i ponadto istnieje różnica potencjałów między punktami dołączania mas, wówczas pojawia się prąd, który płynie przez tworzące pętlę połączenia mas tych części układu (rys. 8.7). W efekcie pojawia się przydźwięk albo przesłuchy. Takie same skutki może wywołać zakłócające pole magnetyczne poprzez zaindukowanie napięcia w przewodzie ekranującym.



Rys. 8.7. Pętla utworzona przez podwójne połączenie uziemień dwóch układów

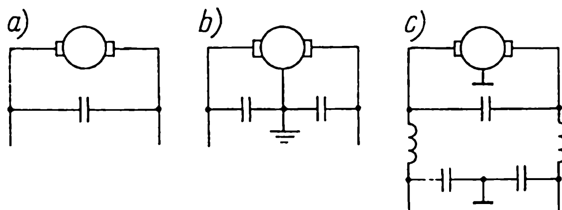
8.4. Usuwanie zakłóceń

Jeśli zależy nam na usunięciu zakłóceń, to trzeba wziąć pod uwagę wszystkie wyżej opisane zagadnienia (p. 8.2). A więc można próbować wyeliminować źródła zakłóceń, zapobiegać ich rozchodzeniu się albo chronić przed tymi sygnałami układy na nie wrażliwe.

Eliminacja źródeł

Jest ona najbardziej pożądana, ale niestety najtrudniejsza do przeprowadzenia, ponieważ źródło zakłóceń najczęściej trudno jest dokładnie zlokalizować, lub — jeśli zakłócenia są przenoszone przez sieć oświetleniową — dostęp do źródła w ogóle nie jest możliwy. Najbardziej eleganckim rozwiązaniem jest zamknięcie źródła zakłóceń w metalowej obudowie. Osłona taka działa najskuteczniej, jeśli jest szczelna ze wszystkich stron (klatka Faradaya), a napięcia stałe są doprowadzone poprzez kondensatory przepustowe.

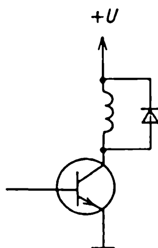
W przypadku silników zaleca się umieszczać elementy blokujące bądź układy filtrujące możliwie blisko wyprowadzeń kolektora. Różne możliwości są pokazane na rys. 8.8. Rozwiązanie najprostsze (a) zawiera jeden kondensator



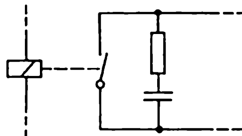
Rys. 8.8. Wytłumianie zakłóceń wytworzonych przez silniki

blokujący (dobierany). Propozycja (b) nadaje się szczególnie do tłumienia zakłóceń współbieżnych, które w takim układzie są zwierane do masy. W układzie (c) zastosowano podwójny filtr typu π, który tłumi zarówno składowe sumacyjne, jak i różnicowe. Wartości kondensatorów (koniecznie bezindukcyjnych) muszą być dobierane doświadczalnie. W połączeniu z indukcyjnościami dławików oraz indukcyjnościami przewodów, pojemności te tworzą obwód rezonansowy, którego częstotliwość drgań powinna leżeć poza pasmem współpracujących układów elektronicznych.

Przy sterowaniu elementów o charakterze indukcyjnym (elektromagnesy, silniki itd.) stosuje się na ogół diodę dołączoną równolegle do obciążenia, która w momencie wyłączenia przejmuje cały prąd i w ten sposób ogranicza skok napięcia na kolektorze tranzystora (rys. 8.9). W przekaźnikach włącza się również równolegle do zestyków dwójniki *RC* (rys. 8.10). Jeśli istnieje taka możliwość (np. ze względu na koszty), to obciążenia indukcyjne powinny być przełączane przy przejściu prądu przez zero.



Rys. 8.9. Dioda zabezpieczająca przed przepięciami w przypadku obciążeń o charakterze indukcyjnym



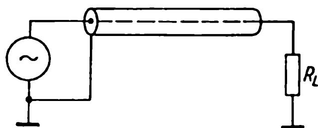
Rys. 8.10. Dwójnik *RC* dołączony do zestyków przekaźnika

Ograniczanie rozchodzenia się zakłóceń

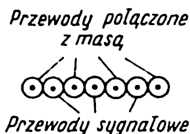
Występują — jak już wspomniano — cztery drogi przenoszenia sygnałów zakłócających. Odpowiednio dla każdej z nich będą podane środki zaradcze.

Sprzężenie pojemnościowe jest najczęściej spotykaną drogą przenoszenia zakłóceń. Uniwersalnym środkiem przeciwko polom elektrycznym jest ekranowanie. Przewód ekranowany powinien być dołączony do masy tylko z jednej strony, w przeciwnym bowiem razie z połączenia mas powstać może pętla (rys. 8.11). Wyjątek stanowią linie transmisyjne wielkiej częstotliwości. W tym przypadku przewód zewnętrzny może być połączony na obu końcach z masą, ponieważ "droga okrężna" utworzona przez chassis[1] lub system przewodów

[1] Metalowa podstawa, na której montuje się elementy układu (przyp. tłum.).



Rys. 8.11. Ekranowanie od pól elektrycznych



Rys. 8.12. Ekranowanie zastępcze stosowane w kablach płaskich

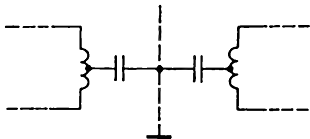
uziemiających nie jest groźna z powodu dużej impedancji dla sygnałów wielkiej częstotliwości. Przy większej liczbie źródeł sygnału należy każde połączenie oddzielnie ekranować i oddzielnie przy każdym źródle uziemić ekran.

Wiązek kablowych należy unikać, chyba że się ułoży "gorące" i "zimne" przewody w oddzielnych wiązkach. Lepsze są w tym przypadku kable płaskie, w których co drugi przewód, albo przynajmniej przewody sąsiadujące z "gorącymi", łączy się z masą (rys. 8.12). Przy większych odległościach zwiększona pojemność między przewodami powoduje "spłaszczenie" zboczy impulsów transmitowanych sygnałów cyfrowych. W najgorszym razie kształt impulsów poprawić można po stronie odbiorczej za pomocą przerzutników Schmitta. Pozostałe niewykorzystane przewody powinny być także przyłączone do masy.

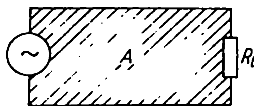
Na płytkach drukowanych pomiędzy ścieżkami sygnałowymi powinny być umieszczone ścieżki dołączone do masy. W ten sposób szkodliwe pojemności pasożytnicze redukowane są 5-krotnie. Także na nie wykorzystanej powierzchni płytki powinno pozostać możliwie dużo nie usuniętej folii miedzianej, którą również łączy się z masą.

Między pierwotnym i wtórnym uzwojeniem transformatora istnieje pasożytnicza pojemność, przez którą zakłócenia — głównie składowa współbieżna — przedostają się z sieci do zasilacza, a stąd do pozostałych układów. Można temu zapobiec używając transformatora z folią ekranującą między uzwojeniami — pierwotnym i wtórnym. Folię tę łączy się z masą (rys. 8.13). Przy ekranowaniu od zakłócających pól magnetycznych powinno przestrzegać się następujących zaleceń:

— odbiornik należy umieścić możliwie daleko od źródła pola magnetycznego;



Rys. 8.13. Eliminacja pasożytniczych pojemności sprzęgających w transformatorze



Rys. 8.14. Wielkość powierzchni A ograniczonej obwodem decyduje o wielkości indukowanego napięcia zakłóceń

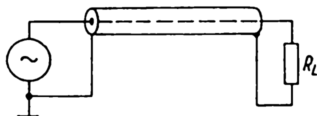
— przewody powinny przebiegać równolegle do linii pola magnetycznego;

— należy unikać prowadzenia masy w postaci pętli, ponieważ z powodu małej oporności takich połączeń mogą wystąpić silne prądy indukowane;

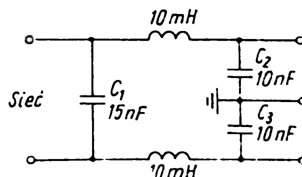
— wartość indukowanego napięcia zależy bezpośrednio od wielkości powierzchni ograniczonej pętlą złożoną z przewodu "sygnałowego" i masy (rys. 8.14); powierzchnia powinna być jak najmniejsza, tzn. oba przewody powinny leżeć możliwie blisko siebie.

Do przesyłania sygnałów o częstotliwościach poniżej 1 MHz doskonale nadają się skręcone pary przewodów, ponieważ — jeśli prądy w obu przewodach są równe — linie pola od jednego zwoju znoszą się nawzajem. Również zakłócenia zewnętrzne znoszą się nawzajem w takiej parze przewodów. W przypadku większej liczby przewodów (np. kabel interfejsu V24), każdy przewód sygnałowy powinien być skręcony z odpowiadającym mu doprowadzeniem masy. W tak wykonanym torze tłumienie składowej współbieżnej zakłóceń dochodzi do 40 dB. W przypadku zakłóceń różnicowych system ten jest mniej przydatny. Większe tłumienie sygnałów zakłócających można uzyskać przez dodatkowe ekranowanie skręconych przewodów.

Przy większych częstotliwościach najlepszym rozwiązaniem jest kabel koncentryczny (do 1 GHz). W przypadku połączenia jak na rys. 8.15, istnieje jedynie pozorna sprzeczność w stosunku do wcześniej postawionego wymagania, żeby ekran był dołączony do masy tylko jednym końcem, tzn. żeby nie płynął przezeń żaden prąd. Przewód zewnętrzny nie pełni tu roli ekranu, natomiast stanowi przewodzącą powierzchnię, na której kończą się linie sił pola elektrycznego pochodzące od przewodu wewnętrznego. Taki układ praktycznie nie wytwarza zakłóceń, jak również jego wrażliwość na zakłócenia zewnętrzne jest minimalna.



Rys. 8.15. Kabel koncentryczny użyty jako zabezpieczenie przed zakłócającym polem magnetycznym



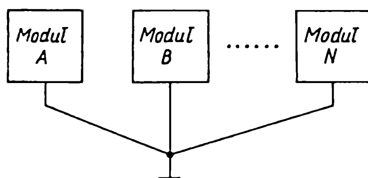
Rys. 8.16. Podwójny filtr typu π

Oddzielne analizowanie środków zabezpieczających przed promieniowaniem elektromagnetycznym nie jest konieczne, gdyż zabezpieczenia od zakłócających pól elektrycznych i magnetycznych stosuje się również przeciwko zakłócającemu promieniowaniu elektromagnetycznemu.

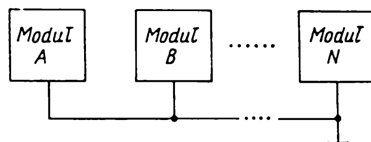
Zagadnienia związane z sygnałami zakłócającymi, które są wprowadzane do układu poprzez sprzężenie galwaniczne, a więc poprzez doprowadzenia, można podzielić na trzy grupy: usuwanie impulsów napięcia przenoszonych przez doprowadzenia sieciowe, usuwanie sygnałów zakłócających wytwarzanych przez impulsowe obciążenie zasilacza oraz poprawne łączenie mas.

Zakłócenie impulsowe pojawiające się na doprowadzeniach napięcia sieciowego można zwalczać za pomocą odpowiednich filtrów. Najskuteczniejsze są podwójne filtry typu π (rys. 8.16), które w gotowej postaci (obudowane) mocuje się w miejscu doprowadzenia kabla sieciowego (tylna ściana obudowy). Kondensator C_1 oraz szeregowo połączone kondensatory C_2 i C_3 tworzą zwarcie dla zakłóceń różnicowych. Zakłócenia współbieżne (sumacyjne) są odprowadzane do masy poprzez C_2 i C_3 (C_1 nie usuwa zakłóceń współbieżnych).

Sygnały zakłócające, które pojawiają się na przewodach doprowadzających napięcia zasilające stałe, a wywoływane są przez procesy przełączania, głównie w cyfrowych stopniach mocy, można usuwać stosując kondensatory ceramiczne bądź tantalowe. Kondensatory zwijane nie wchodzą w rachubę, ponieważ zwnięta folia tworzy szeregowe połączenie indukcyjności i pojemności. Przy pracy impulsowej, a więc przy dużych częstotliwościach, przeważa składowa indukcyjna. Stosuje się również — włączone równolegle do kondensatorów elektrolitycznych zasilacza — kondensatory od 20 nF do 1 µF. Ponadto, na płytkach powinno się umieszczać przynajmniej jeden kondensator ok. 0,1 µF na każde trzy układy scalone. Najlepiej wprost na końcówkach zasilania, aby zmniejszyć wpływ indukcyjności ścieżek doprowadzających. Jeśli nie ma innej możliwości usunięcia zakłóceń powstałych wskutek impulsowych zmian prądu, to stosować należy na poszczególnych płytkach oddzielne scalone stabilizatory napięcia.



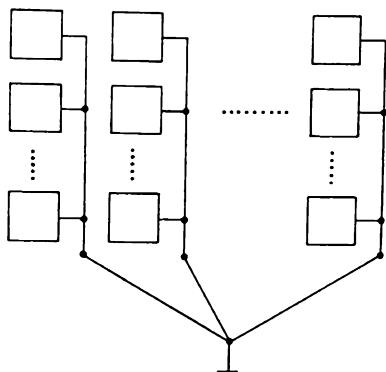
Rys. 8.17. Uziemienia kilku części układu, połączone w postaci gwiazdy



Rys. 8.18. Szeregowe połączenie uziemień kilku części układu

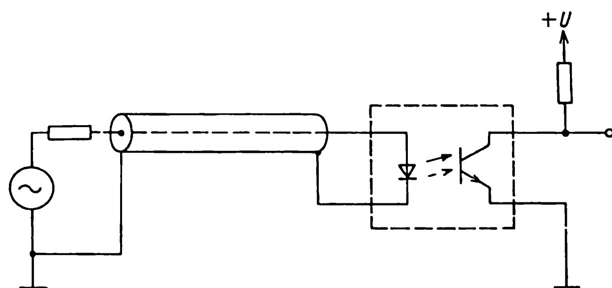
Najlepszą konfiguracją uziemień jest uziemienie w kształcie gwiazdy (rys. 8.17). Uziemienie w postaci szeregowej (rys. 8.18) jest mniej dogodne, ponieważ w tym przypadku sumują się prądy i zakłócenia od poszczególnych stopni układu. Jeśli takiego połączenia nie da się uniknąć, należy przynajmniej zapewnić doprowadzenie wspólnej masy do elementu o największym poborze

mocy (wzmacniacz mocy). W praktyce stosuje się często kombinowane łączenie uziemień (rys. 8.19): połączenia szeregowe na płytkach, a następnie gwiaździste od poszczególnych grup do jednego wspólnego punktu.



Rys. 8.19. Kombinowane połączenie uziemień kilku fragmentów układu

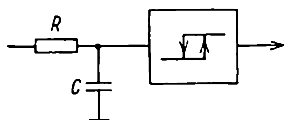
W przypadku długich przewodów sygnałowych — ze względu na różnice potencjałów mas oraz istnienie pól zakłócających — nie można bezpośrednio zastosować sprzężenia galwanicznego. W takiej sytuacji oddziela się galwanicznie obie części układu, np. za pomocą transoptora (rys. 8.20) (dawniej stosowano transformatory separujące i przekaźniki).



Rys. 8.20. Zastosowanie transoptora do rozdzielenia potencjałów

Odbiornik

Jeśli chodzi o zakłócenia w odbiorniku, to niewiele można zdziałać. Można go ekranować stosując zamkniętą obudowę. Materiał o dużej przewodności chroni jedynie przed polem elektrycznym. Pola magnetyczne wymagają stosowania materiałów o dużej przenikalności magnetycznej jak stal lub permaloj. Zakłócające sygnały wielkoczęstotliwościowe nałożone na sygnały użyteczne, mogą być odfiltrowane na wejściach układu za pomocą filtrów dolnoprzepustowych. W



Rys. 8.21. Filtr dolnoprzepustowy z układem formującym sygnał o stromym zboczu

przypadku sygnałów cyfrowych zbocza impulsów mogą przy tym zostać tak mocno spłaszczone, że trzeba je będzie ponownie formować, stosując przerzutniki Schmitta (rys. 8.21). Powinno się unikać elementów wrażliwych na zakłócenia np. przerzutników monostabilnych. Inna możliwość polega na stosowaniu specjalnych, powolnych układów logicznych, odpornych na zakłócenia.

Literatura

1. Applikationsvorschlag der Firma Ferranti für den ZN 425 E.
2. Dittmar E.: *Mikrocomputer–Einsatz in der Automatisierung*. Würzburg Vogel, 1979.
3. Klein R.D.: Drei integrierte IEC-Bus-Controller im Vergleich. *Elektronik*, 1982, H. 12, s. 64.
4. Luxmeter für 100 bis 100000 Lux. *Markt und Technik*, 1980, Nr. 41, s. 16.
5. Rüttger M.: Programmierbarer RC-Oszillator. *Elektronik*, 1981, H. 6, s. 127–129.
6. Siemens-Datenbuch. Mikrocomputer-Bausteine System SAB 8080.

Skorowidz

Czytelnikom zainteresowanym techniką mikroprocesorową polecamy książkę K. Sachy, P. Misiurewicza i T. Kręglewskiego pt. „Przewodnik po technice mikrokomputerowej", która wkrótce ukaże się w księgarniach. W książce tej omówiono podstawowe zagadnienia związane z budową, zasadami działania i oprogramowaniem mikrokomputerów. Czytelnik znajdzie tam między innymi wiadomości dotyczące: różnych typów mikroprocesorów, ich elementów, zasad działania i parametrów; pamięci mikrokomputerów, urządzeń zewnętrznych, sprzęgów; oprogramowania, w tym systemów operacyjnych, programów usługowych, wybranych języków programowania; zastosowania mikrokomputerów w różnych dziedzinach, np. w systemach czasu rzeczywistego, do sterowania urządzeń, do przetwarzania tekstów i inne.
Cały materiał „Przewodnika" zebrano w postaci haseł, w których są omówione poszczególne zagadnienia lub grupy zagadnień. Podano także odpowiedniki angielskie, francuskie, niemieckie i rosyjskie użytych terminów.

Mikrokomputery

Książki serii „Mikrokomputery" mają na celu prezentację i systematyzację wiedzy z zakresu systemów mikrokomputerowych (mikroprocesorowych). Na ich podstawie będzie możliwe: projektowanie, konstruowanie, modyfikowanie, naprawianie, rozszerzanie i adaptowanie do określonych zastosowań systemów mikrokomputerowych. Problematyka serii obejmuje: elementy dyskretne i układy scalone cyfrowe stosowane w systemach mikrokomputerowych, metodologię projektowania i podstawowe algorytmy syntezy układów cyfrowych; rodziny mikroprocesorowe od strony architektury, jak i oprogramowania; układy sprzęgające mikrokomputery z urządzeniami zewnętrznymi, wskazówki praktyczne do projektowania sprzętu mikrokomputerowego, narzędzia i metodologię projektowania oprogramowania oraz zastosowanie systemów mikrokomputerowych.
Książki serii są przeznaczone dla szerokiego kręgu Czytelników – od niespecjalistów „wchodzących" w dziedzinę systemów mikrokomputerowych (informatycznych) do specjalistów poszukujących szczegółowych informacji o wskazanych problemach.

Mikrokomputery

W serii ukazały się następujące pozycje:

J. Bielecki – Język C – interpretacja standardu
J. Bielecki – System operacyjny ISIS-II
J. Bielecki – Turbo Pascal wersja 3.0
J. Boisgontier, S. Brébion – Basic dla wszystkich
W. Iszkowski – Nauka programowania w języku BASIC dla początkujących
M. Kalinowska-Iszkowska, W. Iszkowski – Klucze do Basicu
J. Karczmarczuk – Mikroprocesor Z80

W najbliższym czasie ukażą się:

J. Bielecki – Fortran 77
J. Bielecki – Język Forth
J. Bielecki – Język Turbo C
J. Bielecki – Turbo Pascal z grafiką dla IBM PC
J. Bielecki – Wprowadzenie do języka C
L. Bułhak, R. Goczyński, M. Tuszyński – System operacyjny MS-DOS, PC-DOS
W. Cellary, Z. Królikowski – Wprowadzenie do projektowania baz danych – dBase III
W. Cellary, J. Rykowski – System operacyjny CP/J dla ELWRO 800 Junior
W. Cellary, W. Wieczerzycki – Wielozadaniowy system operacyjny czasu rzeczywistego iRMX-88
R. Świniarski – System operacyjny CP/M

WYDAWNICTWA NAUKOWO-TECHNICZNE
ul. Mazowiecka 2/4, 00-048 Warszawa
tel. 26-72-71 do 79
Dział Upowszechniania i Sprzedaży
tel. 27-56-87

WNT Warszawa 1988 r.
Wyd. I. Nakład 19 700 + 300 egz.
Ark. wyd. 10.0. Ark. druk. 10.0. Format B5.
Podpisano do druku w maju 1988 r.
Druk ukończono w maju 1988 r.
Symbol Et/82196/WNT

Olsztyńskie Zakłady Graficzne Zam. 637/88
U-61/75

Cena zł 490,-

ISBN 83-204-0934-9